

# Spezifikationen zu Strukturabfragen

2027 V02

## Allgemeine Technische Dokumentation

## Impressum

Titel	Spezifikationen zu Strukturabfragen. Allgemeine Technische Dokumentation. 2027 V02
Veröffentlichung	30. Juni 2026

### Auftrag

---

Auftraggeber	Gemeinsamer Bundesausschuss (G-BA)
--------------	------------------------------------

### Herausgeber

---

IQTIG – Institut für Qualitätssicherung und Transparenz im Gesundheitswesen  
Katharina-Heinroth-Ufer 1 10787 Berlin  
info@iqtig.org | www.iqtig.org | (030) 58 58 26-0

# Informationen zum Bericht

## BERICHTSDATEN

---

### **Spezifikationen zu Strukturabfragen. Allgemeine Technische Dokumentation. 2027 V02**

Datum der Veröffentlichung      30. Juni 2026

## AUFTRAGSDATEN

---

Auftraggeber                      Gemeinsamer Bundesausschuss (G-BA)

# Inhaltsverzeichnis

Impressum.....	2
Informationen zum Bericht .....	3
Inhaltsverzeichnis .....	4
Tabellenverzeichnis.....	8
Abbildungsverzeichnis .....	10
Abkürzungsverzeichnis .....	11
Änderungsindex.....	13
Leseanleitung .....	14
1    Einleitung.....	15
1.1    Spezifikationsbegriff .....	16
1.2    Zielsetzung und Zielgruppe .....	17
A Prozesse.....	18
2    Auslösung.....	19
3    Datenerfassung .....	20
3.1    Gestaltung von Eingabemasken .....	20
3.2    Einrichtungsidifizierende Daten .....	23
3.3    QS-Daten .....	24
3.4    Plausibilitätsprüfungen .....	24
4    Exportdaten.....	27
4.1    Erzeugen der Exportdatei .....	27
4.2    Datenprüfung.....	29
4.3    Beispiele für Exportdateien.....	30
4.4    Umgang mit Umlaut-Domains und E-Mail-Adressen .....	30
5    Datenübermittlung.....	31
5.1    Allgemeine Vorgaben.....	32
5.2    Eindeutige Kennzeichnung der XML-Exportdateien .....	32
5.2.1    Identifizierung von Datensätzen anhand von Vorgangsnummern.....	33
5.2.2    Benennung der Exportdateien.....	34
5.2.3    Abgrenzung von Test- und Regelbetrieb .....	37



5.3	Datenversand.....	40
5.3.1	Datenversand via E-Mail von den Krankenhäusern an das IQTIG.....	40
5.3.2	Datenversand via E-Mail von den Krankenhäusern an die Landesaufsichtsbehörden .....	41
5.3.3	Datenversand über das E-Mail-Verfahren gemäß GGT an die LVKK/EK.....	41
5.4	Datenflüsse.....	41
5.4.1	Datenfluss der QS-Daten an das IQTIG.....	41
5.4.2	Datenfluss der QS-Daten an die Landesverbände der Krankenkassen und Ersatzkassen .....	42
6	Rückprotokollierung.....	49
6.1	Funktion, Aufbau und Erstellung.....	50
6.1.1	Funktion des Datenflussprotokolls im Datenfluss .....	50
6.1.2	Aufbau und Erstellung .....	50
7	Prozesse in den Datenannahmestellen .....	57
7.1	Verarbeitungsschritte ohne Verwendung der Spezifikationskomponente Datenprüfprogramm .....	58
7.2	Verarbeitungsschritte bei Anwendung der Spezifikationskomponente Datenprüfprogramm .....	65
7.3	Hinweise.....	67
B	Komponenten .....	70
8	Spezifikationskomponenten .....	71
8.1	Spezifikationskomponenten .....	71
8.1.1	Datenbank .....	71
8.1.2	Schema .....	71
8.1.3	Ausfüllhinweise.....	71
8.1.4	Dokubögen.....	72
8.1.5	TechDok .....	72
8.1.6	ÜbersichtÄnderungen .....	72
8.1.7	Komponentenübersicht.....	72
8.1.8	Datenprüfprogramm .....	72
8.2	Weitere spezifikationsrelevante Dateien und Programme.....	73
8.2.1	Verschlüsselungsprogramm .....	73

8.2.2	Datenbank zu Datenserviceinformationen .....	73
8.2.3	Excel-Datei zur optionalen Anwendung beim Verfahren nach PPP-RL .....	73
9	QS-Dokumentation.....	74
9.1	Anmerkungen zur Struktur der Spezifikation zur QS-Dokumentation.....	74
9.1.1	Tabellenstruktur der Datenbanken .....	74
9.1.2	Abfragen der Datenbank .....	76
9.2	Datenfeldbeschreibung .....	78
9.2.1	Dokumentationsmodule (Datensätze) .....	80
9.2.2	Teildatensätze.....	80
9.2.3	Datenfelder (Bogenfelder).....	83
9.2.4	Überschriften vom Dokumentationsbogen.....	91
9.2.5	Ausfüllhinweise.....	91
9.3	Plausibilitätsprüfungen .....	92
9.3.1	Die Regeltabelle.....	93
9.3.2	Regelsyntax .....	94
9.3.3	Funktionen.....	100
9.3.4	Syntaxvariablen .....	102
9.3.5	Einzelregeln .....	102
9.3.6	Teildatensatzübergreifende Regeln.....	102
9.3.7	Regeln mit Teildatensatz-Listenfeldern.....	103
9.3.8	Feldgruppenregeln .....	103
9.3.9	Prüfung von Feldeigenschaften .....	109
9.3.10	Verfahren für die Evaluation von Regeln .....	112
9.4	Listen von Schlüsselkodes (OPS;ICD) .....	113
9.4.1	OPS-Listen .....	113
9.4.2	ICD-Listen.....	114
9.4.3	Relationstabellen ICD- und OPS-Listen.....	114
9.5	Exportfeldbeschreibung .....	115
9.5.1	Exportmodule .....	115
9.5.2	Exportdatensatz.....	116
9.6	Versionierung.....	121

9.6.1	Grundlegende Definitionen .....	121
9.6.2	Delta-Informationen zur vorhergehenden Version.....	122
9.6.3	Abgrenzung zwischen Spezifikationsjahren und Datensatzformaten.....	124
9.6.4	Version des Exportverfahrens .....	124
9.7	Administrative Objekte .....	124
9.7.1	XML-Mapping in der Spezifikationsdatenbank (QSDOK) .....	125
9.7.2	Datenservices .....	126
9.7.3	Prüfschritte .....	127
10	XML-Schema .....	129
11	Tools .....	130
11.1	Datenprüfprogramm .....	131
11.1.1	Umfang der Prüfungen .....	131
11.1.2	Programmaufruf .....	131
11.1.3	Verzeichnisstruktur .....	133
11.1.4	Ausgabe.....	134
11.1.5	Grafische Oberfläche .....	134
C	Anhang .....	136
	Glossar .....	136

## Tabellenverzeichnis

Tabelle 1: Informationen aus der Datenbank, welche im GUI verwendet werden .....	20
Tabelle 2: Arten der Plausibilitätsprüfungen .....	25
Tabelle 3: XML-Schemata zur Prüfung vor der Verschlüsselung am Beispiel von QSFFx .....	29
Tabelle 4: Benennungselemente der Exportdateien.....	34
Tabelle 5: Ausfüllen der Elemente eines Validation-Items in Abhängigkeit von den Fehlerarten	52
Tabelle 6: Beispiele von Fehlermeldungen .....	53
Tabelle 7: Struktur der Tabelle <code>ExportFormatExportModul</code> .....	68
Tabelle 8: Struktur der Tabelle <code>Modul</code> .....	80
Tabelle 9: Struktur der Tabelle <code>Bogen</code> .....	81
Tabelle 10: Inhalte der Tabelle <code>BogenTyp</code> .....	82
Tabelle 11: Struktur der Tabelle <code>BogenFeld</code> .....	84
Tabelle 12: Struktur der Tabelle <code>Feld</code> .....	86
Tabelle 13: Struktur der Tabelle <code>BasisTyp</code> .....	88
Tabelle 14: Struktur der Tabelle <code>Schluessel</code> .....	88
Tabelle 15: Struktur der Tabelle <code>SchluesselWert</code> .....	90
Tabelle 16: Struktur der Tabelle <code>Abschnitt</code> .....	91
Tabelle 17: Arten von Hinweistypen.....	92
Tabelle 18: Tabelle <code>RegelTyp</code> .....	92
Tabelle 19: Struktur der Tabelle <code>Regeln</code> .....	93
Tabelle 20: Struktur der Tabelle <code>RegelFelder</code> .....	94
Tabelle 21: Struktur der Tabelle <code>MehrfachRegel</code> .....	94
Tabelle 22: Basistypen der Datenfelder in den Plausibilitätsregeln .....	95
Tabelle 23: Präzedenz und Assoziativität der Operatoren .....	97
Tabelle 24: Typen von <code>Feldgruppen</code> .....	103
Tabelle 25: Struktur der Tabelle <code>FeldGruppe</code> .....	104
Tabelle 26: Struktur der Tabelle <code>FeldGruppeFelder</code> .....	105
Tabelle 27: Formale Definition einer <code>Feldgruppe</code> .....	107
Tabelle 28: Identitätsprüfung zwischen dokumentierten OPS-Kodes und Kodes von OPS- Listen .....	114
Tabelle 29: Struktur der Tabelle <code>ExportModul</code> .....	115
Tabelle 30: Struktur der Tabelle <code>ZusatzFeld</code> .....	118
Tabelle 31: Struktur der Tabelle <code>Ersatzfeld</code> .....	119
Tabelle 32: Struktur der Tabelle <code>ErsatzFuerFeld</code> .....	120
Tabelle 33: Struktur der Tabelle <code>DeltaNeu</code> .....	122
Tabelle 34: Struktur der Tabelle <code>DeltaAttribut</code> .....	123
Tabelle 35: Struktur der Tabelle <code>DeltaGeloescht</code> .....	123

Tabelle 36: Inhalt der Tabelle TabellenFeldStruktur (fkTabellenFeldStruktur  
= Regeln) .....124

Tabelle 37: Felder der Abfrage vPruefung ..... 127

## Abbildungsverzeichnis

Abbildung 1: Kommunikationsbeziehungen zwischen Leistungserbringer, Landesaufsichtsbehörden, Landesverbänden der Krankenkassen und Ersatzkassen sowie dem IQTIG inkl. Richtlinienvorgaben.....	31
Abbildung 2: Datenflüsse im Test- und Regelbetrieb .....	37
Abbildung 3: Übersicht direkter Datenfluss an die BAS .....	41
Abbildung 4: Datenfluss der Rückprotokollierung .....	42
Abbildung 5: Attribut "originator" im Prüfungs- und Fehlerprotokoll .....	50
Abbildung 6: Tabellen und Relationen der Datenfelddbeschreibung .....	79
Abbildung 7: Tabelle OPSRelation .....	115
Abbildung 8: Tabelle Datenservice.....	126
Abbildung 9: Beispiel einer typischen Verzeichnisstruktur.....	134
Abbildung 10: Grafische Oberfläche des Datenprüfprogramms .....	135

## Abkürzungsverzeichnis

Abkürzung	Bedeutung
AES	Advanced Encryption Standard (Verschlüsselungsalgorithmus)
AG	Arbeitsgruppe
AIS	Arztinformationssystem
ASCII	American Standard Code for Information Interchange (Amerikanischer Standard-Code für den Informationsaustausch)
BAS	(Bundes-)Auswertungsstelle
BSI	Bundesamt für Sicherheit in der Informationstechnik
CSV	Comma-separated values (Dateiformat)
DAS	Datenannahmestelle
DB	Datenbank
DIMDI	Deutsches Institut für Medizinische Dokumentation und Information
Dv	direkte Verfahren
DPP	Datenprüfprogramm
DÜV	Datenübermittlungsvereinbarung
FAB	Fachabteilung
G-BA	Gemeinsamer Bundesausschuss
GKV	Gesetzliche Krankenversicherung
GUID	Globally Unique Identifier
HTML	Hypertext Markup Language (Hypertext-Auszeichnungssprache)
ID	Identifikationsnummer
IK	Institutionskennzeichen
IKNR	Institutionskennzeichennummer
IQTIG	Institut für Qualitätssicherung und Transparenz im Gesundheitswesen
JRE	Java Runtime Environment (Java-Laufzeit-Umgebung)
JVM	Java Virtual Machine, ist Teil der Java-Laufzeitumgebung
K	Kann-Feld
KH	Krankenhaus, Krankenhäuser
KK	Krankenkassen
KIS	Krankenhausinformationssystem

Abkürzung	Bedeutung
LAB	Landesaufsichtsbehörde/n
LVKK	Landesverbände der Krankenkassen
LE	Leistungserbringer
LID	Leistungserbringeridentifizierenden Daten
M	Muss-Feld
OPS	Operationen- und Prozedurenschlüssel
OR	ODER-Operator
PPP-RL	Richtlinie zur Personalausstattung in Psychiatrie und Psychosomatik
Qb-R	Regelungen zum Qualitätsbericht der Krankenhäuser
QFR-RL	Qualitätssicherungs-Richtlinie Früh- und Reifgeborene
QS	Qualitätssicherung
QSDOK	Access-Datenbank, in der die Dokumentationen für Strukturabfragen spezifiziert werden
QSFFx-RL	Richtlinie zur Versorgung der hüftgelenknahen Femurfraktur
RL	Richtlinie
RSA	Verfahren zur Datenverschlüsselung, entwickelt von R. Rivest, A. Shamir und L. Adleman
SGB	Sozialgesetzbuch
SGB V	Sozialgesetzbuch Fünftes Buch
SWA	Softwareanbieter
TB	Testbetrieb
TDS	Teildatensatz
TPacker	Programm für die Transportverschlüsselung
URL	Uniform Resource Locator (einheitlicher Ressourcenzeiger)
V	Versionierung
vdek	Verband der Ersatzkassen
XML	Extensible Markup Language
XPacker	Verschlüsselungsprogramm
XSD	XML-Schema-Datei
XSLT	Extensible Stylesheet Language Transformation (Programmiersprache zur Transformation von XML-Dokumenten)
ZIP	zipper, Abkürzung für ein Format für verlustfrei komprimierte Dateien



## Änderungsindex

Änderungen der Datenbanken im Vergleich zur Vorversion lassen sich anhand der Delta-Tabellen mit dem Präfix „Delta“ nachvollziehen. Ohne Version im Tabellennamen bezieht sich die Deltatable auf die vorherige Version. Anderenfalls wird jeweils das Delta zur genannten Version aufgelistet.

### Kapitelübergreifende Änderungen:

- Erfassungsjahr wird ersetzt durch Spezifikationsjahr
- Konkretisierungen und Optimierung von Formulierungen
- Anpassung von Jahreszahlen, Beispielen, Abbildungen und Tabellen
- Korrektur von Fehlern und Ergänzung von fehlenden Inhalten
- Anpassung von Abkürzungen

Konkrete Informationen zu den inhaltlichen Änderungen sind der Spezifikationskomponente Übersicht Änderungen bzw. den aktuellen Beschlüssen des G-BA zu entnehmen.

Die spezifischen Änderungen der vorliegenden Technischen Dokumentation werden im Folgenden mit Bezug zur jeweiligen Version dargestellt.

Änderung	Kapitel/Abschnitt	Version
Redaktionell, ergänzende Aufnahme von Inhalten zur Spezifikation gemäß QFR-RL zusätzlich zu den bestehenden Angaben von FFx und PPP		2027 V01
Anpassung bzgl. nicht vorhandener TDS im Abschnitt „Verfahren für die Evaluation von Regeln“	9.3.10	2027 V02

## Leseanleitung

Diese Technische Dokumentation orientiert sich in ihrem Aufbau an den Abläufen der Erfassung und Übermittlung der Qualitätssicherungsdaten. Ziel dieser Struktur ist es, eine nachvollziehbare und logische Sicht auf die Umsetzung und Durchführung der beschriebenen Schritte zu gewährleisten. Die Prozesse und Unterprozesse werden im **Abschnitt A** beschrieben und spiegeln die reale, chronologische Abfolge wider. Jede Prozessbeschreibung berücksichtigt zudem die unterschiedlichen Komponenten, die für die Umsetzung benötigt und in **Abschnitt B** detailliert beschrieben werden. Im **Abschnitt C** wird ein Glossar mit den wichtigen Begriffen zum Themenbereich der Spezifikation zur Verfügung gestellt.

Für eine korrekte Umsetzung der Spezifikation ist es notwendig, die Dokumentation entsprechend ihrer Anordnung von Prozessen zu Komponenten zu befolgen. Einige Bereiche, die sich ausschließlich an bestimmte Zielgruppen richten, sind entsprechend gekennzeichnet.

### Legende

Die in dieser Dokumentation verwendeten Symbole heben bestimmte Aspekte bei der Umsetzung der Spezifikation hervor.



#### Achtung

Beschreibt Ursache, Folge und Vermeidung einer besonderen Fehlanwendung, die zu Problemen bei der Implementierung oder Ähnlichem führen kann.

---



#### Hinweis

Nützliche Informationen, Tipps oder Ratschläge zur Anwendung. Keine wesentlichen oder für das korrekte Funktionieren erforderlichen Informationen.

---

### Beispiel:

Beispiele sind ein Hilfsmittel, um zuvor vermittelte Informationen oder konkrete Abschnitte der Anwendung zu verdeutlichen.

---

# 1 Einleitung

Diese Technische Dokumentation beschreibt die Spezifikationen gemäß folgender Richtlinien:

Die **Richtlinie über Maßnahmen zur Qualitätssicherung der Versorgung von Früh- und Reifgeborenen (QFR-RL)** definiert ein Stufenkonzept der perinatologischen Versorgung in Krankenhäusern. Sie regelt verbindliche Mindestanforderungen an die Versorgung von Früh- und Reifgeborenen und Zuweisungskriterien von Schwangeren nach dem Risikoprofil der Schwangeren oder des Kindes.

- Die **Richtlinie zur Personalausstattung Psychiatrie und Psychosomatik-Richtlinie (PPP-RL)** gemäß § 136a Absatz 2 Satz 1 SGB V, die die Ausstattung stationärer psychiatrischer und psychosomatischer Einrichtungen mit notwendigem therapeutischem Personal regelt. Diese Spezifikationen sind vom Gemeinsamen Bundesausschuss (G-BA) noch nicht verabschiedet worden.
- Die **Richtlinie zur Versorgung der hüftgelenknahen Femurfraktur (QSFFx-RL)** gemäß § 136 Absatz 1 Satz 1 Nummer 2 SGB V für zugelassene Krankenhäuser. Sie beschreibt auch die spezifischen Nachweise, die Krankenhäuser erbringen müssen, um die Mindestanforderungen zu erfüllen. Diese Nachweise sind durch eine Checkliste, die auf einer Strukturabfrage basiert, am Stichtag den Landesverbänden der Krankenkassen und den Ersatzkassen vorzulegen.

Das vorliegende Dokument richtet sich an die Leistungserbringer (LE) und an die mit der Umsetzung der Spezifikation beauftragten Softwareanbieter (SWA) und beschreibt dabei die relevanten Prozesse und Komponenten. In Ergänzung zu diesem Dokument gibt es je Richtlinie ein Zusatzdokument, welches richtlinien-spezifische Informationen und Charakteristika enthält („Verfahrensspezifische Technische Dokumentation“). Diese gilt es ebenfalls für eine spezifikationskonforme Umsetzung verpflichtend zu beachten.

Auf den folgenden Seiten kommt es zur Erklärung des Begriffes „Spezifikation“ sowie zur Adressierung der Zielsetzung und Zielgruppe. Im Weiteren wird dann unter A Prozesse (ab Seite 18) detailliert auf die Prozesse bzw. Prozessschritte zur Umsetzung der Spezifikation eingegangen, wobei alle für die Leistungserbringer bzw. ihre Softwareanbieter relevanten Prozesse beschrieben werden. Diese Prozessbeschreibungen dienen der näheren Erläuterung der erforderlichen Schritte beim Leistungserbringer und sind als verbindliche Handlungsanleitung zu betrachten. Damit soll erreicht werden, dass alle Leistungserbringer die Komponenten korrekt anwenden, Dokumentationspflichten erkennen und Klarheit darüber besteht, wie Datenlieferungen zu verschlüsseln und an welche Datenannahmestelle (DAS) versendet werden muss. Der Bereich B Komponenten (ab Seite 70) beschreibt die einzelnen Komponenten. Die wesentliche Komponente ist die Datenbank, die die Grundlage zur Erstellung der Softwareprodukte bildet. Auch die weiteren Komponenten enthalten Vorgaben bzgl. der Softwareprodukte und zur Umsetzung der Prozesse. In der Komponentenbeschreibung werden unter anderem Struktur, Funktionsweise und

Inhalte der Datenbanken, des XML-Schemas und der weiteren Tools (z. B. Datenprüfprogramm) erläutert.

Zusätzlich zu dem vorliegenden Dokument stehen auf der Website des IQTIG Informationen für Endanwender zu den einzelnen Verfahren und zur Erleichterung der Dokumentation bereit. Zu Letzterem gehören die Dokumentationsbögen und Ausfüllhinweise. Diese Dokumente, die sich an Leistungserbringer richten, die Anwender der Software sind (z. B. Ärztinnen oder Ärzte), sind unter Berücksichtigung verschiedener Anforderungen möglichst anwenderorientiert und verständlich formuliert. Neben der Verständlichkeit werden beispielsweise auch Aspekte wie Einheitlichkeit, technische Umsetzbarkeit und Aufwand bei Verfahrensteilnehmern berücksichtigt.



#### **Achtung**

Die verbindlichen Vorgaben der Spezifikationen sind einzuhalten. Die Softwareprogramme müssen die Erfassung aller Daten gemäß den jeweils relevanten Richtlinien und dieser daraus abgeleiteten Spezifikation ermöglichen, diese Daten auf Vollständigkeit und Plausibilität prüfen, im vorgegebenen Format an die datenentgegennehmenden Stellen exportieren und fehlerhafte (von den Datenstellen abgelehnte) Datensätze in geeigneter Weise dem Anwender zur Korrektur und zum erneuten Export vorlegen. Darüber hinaus bietet die Spezifikation den Softwareanbietern Hinweise für die Gestaltung der Benutzeroberfläche und die benutzerfreundliche Plausibilitätsprüfung unmittelbar bei der Dateneingabe.

---

## **1.1 Spezifikationsbegriff**

Die Spezifikation ist die Gesamtheit aller Vorgaben bezogen auf ein Spezifikationsjahr, nach denen die Dokumentation selbst und die Übermittlung der Daten erfolgen muss.

Um die komplexen Anforderungen an die Dokumentation für Strukturabfragen sowie die zugehörigen Datenflüsse zu erfüllen, besteht die Spezifikation aus verschiedenen Komponenten. Als Komponenten werden dabei Access-Datenbanken, die Technische Dokumentation, Ausfüllhinweise und anderes bezeichnet. Jeder Anwender bekommt damit das für ihn Relevante in einem eigenen Spezifikationspaket als Download zur Verfügung gestellt. Jedes dieser Pakete kann auf diese Weise auch unabhängig von den anderen aktualisiert werden. Für eine spezifikationskonforme Umsetzung von Softwareprodukten sind alle Spezifikationskomponenten zu berücksichtigen.

Sowohl die Spezifikationspakete als auch die einzelnen Komponenten werden nach einem einheitlichen Schema benannt, das bereits im Namen übersichtlich die relevanten Informationen wie Betriebsart, Exportformat und Versionierung enthält. Dieses Schema wird in einem separaten Dokument der Spezifikation detailliert erläutert. Durch die Versionierung sowohl auf der Ebene der Pakete als auch auf der Ebene der Komponenten ist gewährleistet, dass der aktuelle Stand leicht ersichtlich ist. Zudem wird die Kommunikation über die anzuwendenden Bestandteile der Spezifikation erleichtert.

Jedem Paket liegt eine Auflistung der einzelnen Komponenten und ggf. eine Übersicht über die Änderungen zur vorhergehenden Version bei.

## 1.2 Zielsetzung und Zielgruppe

Die Spezifikation ist ein komplexes Regelwerk, das mithilfe verschiedener Komponenten verbindliche Grundlagen für alle Prozesse im Zusammenhang mit der Erfassung und Übermittlung von Daten bei den Leistungserbringern vorgibt und beschreibt. Die Komponenten der Spezifikation sind daher so ausgestaltet, dass sie von QS- und/oder IT-/EDV-Expertinnen und -Experten verstanden werden. Die Spezifikation richtet sich ausschließlich an diesen Teilnehmerkreis. Die Regelung und die Art der Darlegung der Spezifikationskomponenten sind auf eine möglichst automatisierte Nutzung durch diesen Personenkreis ausgerichtet.

Externe Qualitätssicherungsmaßnahmen, die eine Erhebung der strukturellen Voraussetzungen zum Ziel haben, stellen eine Reihe von Anforderungen an die Datenerhebung, Datenerfassung und Plausibilitätsprüfung, um valide, reliable und vergleichbare Daten gewinnen zu können. Die Erfassung und Plausibilitätsprüfung durch unterschiedliche Softwareumsetzungen beinhaltet grundsätzlich die Gefahr einer Verzerrung der Daten.

Die Vorgaben der Spezifikation, die eine einheitliche Festlegung von Datenfeldbeschreibungen, Plausibilitätsregeln, Grundsätzen der Benutzerschnittstellengestaltung und Datenübermittlungsformaten umfassen, sollen dazu dienen, dieser Gefahr entgegenzuwirken. Dadurch wird die Erhebung valider und vergleichbarer Daten sowie ein unter datenschutzrechtlichen Gesichtspunkten sicherer Datenfluss gewährleistet.

## A Prozesse

In diesem Kapitel werden die Prozessschritte sowie die in jedem Prozessschritt benötigten Werkzeuge der Dokumentation in Bezug auf die Erfassung, Verarbeitung und Datenübermittlung beschrieben.

## 2 Auslösung

Eine automatische Auslösung der Dokumentationspflicht gibt es bei den Spezifikationen gemäß den Richtlinien QSFFx, PPP und QFR nicht. Die Dokumentation liegt in der eigenen Verantwortung jedes Leistungserbringers. Sie ist jedoch Voraussetzung für die Abrechnung von Leistungen nach QSFFx-RL und QFR-RL, ebenso erfolgen Abschläge auf die Vergütung nach PPP-RL, wenn nicht, nicht vollständig oder nicht fristgerecht dokumentiert wurde. Weitere Informationen über zum Beispiel richtlinienspezifische Exportfristen, Gültigkeit, Erinnerungswesen etc. sind in den jeweiligen Zusatzdokumenten („Verfahrensspezifische Technische Dokumentation“) zu finden.

Es wird für das QSFFx Nachweisverfahren jedoch eine generelle Auslösung des Bogens jährlich zum 15.11. des Spezifikationsjahresempfohlen, um den Leistungserbringer an seine Dokumentationspflichten im Rahmen der QSFFx-RL zu erinnern. Zusätzlich wird eine Erinnerung zwei Wochen vor Ablauf der Datenlieferfrist (31.12. des Spezifikationsjahres) empfohlen, insofern bis dato kein Export im Rahmen des Nachweisverfahrens erfolgt ist. Beim Verfahren nach QFR-RL wird ebenfalls empfohlen, die Anwender bzgl. der Dokumentation und Übermittlung der Strukturabfrage, die sich auf das vorherige Erfassungsjahr bezieht, zum 01.01.EJ+1 zu erinnern.



### Hinweis

Eine fallbezogene bzw. automatische Auslösung, wie es in der Basisspezifikation der Fall ist, gibt es bei den Strukturabfragen nicht.

---

### 3 Datenerfassung

Beim Prozess der Erfassung werden Daten entweder automatisch aus dem Krankenhausinformationssystem/Arztinformationssystem (KIS/AIS) in die Eingabemaske der Erfassungssoftware übertragen oder durch den Dokumentierenden manuell erfasst.

Neben QS-Daten sind auch einrichtungsidentifizierende Daten wie die QS-relevante Standortnummer für die Leistungserbringer zu dokumentieren. Die Erfassung ist abgeschlossen, wenn alle zu dokumentierenden Datenfelder unter Berücksichtigung von Abhängigkeiten und Plausibilitätsprüfungen vollständig erfasst wurden (Abschnitt 3.4).

Als Vorlage für die Gestaltung der Eingabemaske (Abschnitt 3.1) durch den Softwareanbieter dient die Datenfeldbeschreibung des jeweiligen Moduls (Abschnitt 9.2).

#### 3.1 Gestaltung von Eingabemasken

Die Benutzeroberfläche einer Erfassungssoftware (Graphical User Interface = GUI) soll ergonomisch und anwenderfreundlich gestaltet sein. Gestaltung und Layout der Eingabemaske sind Aufgabe der Softwareanbieter. Neben Anforderungen der Kunden werden üblicherweise firmeninterne Standards bzw. Vorgaben des Betriebssystems (z. B. Windows) für das „look and feel“ berücksichtigt.



##### Hinweis

Als Referenz für die sichtbaren Inhalte dienen die Dokumentationsbögen, die als Bestandteil der Spezifikation durch das IQTIG veröffentlicht werden. Die Dokumentationsbögen werden als PDF-Dokumente bereitgestellt, die aus der Spezifikationsdatenbank automatisch generiert worden sind. Bei den Dokumentationsbögen handelt es sich um Formulare zur Ansicht (Muster), die nicht zur Dokumentation zu verwenden sind.

Tabelle 1 gibt einen Überblick darüber, welche Informationen der Spezifikationsdatenbank (identifiziert durch Tabelle und Attribut) bei der Erstellung der Dokumentationsbögen berücksichtigt werden und somit auch in den Erfassungssystemen sichtbar sein sollen.

Tabelle 1: Informationen aus der Datenbank, welche im GUI verwendet werden

Tabelle	Attribut	Bemerkung	sichtbar für Anwender
Modul	name	Kürzel des Datensatzes (z. B. FFX), erscheint üblicherweise im Titel des Formulars	ja
Modul	bezeichnung	Bezeichnung des Datensatzes (z. B. Qualitätssicherungs-Richtlinie zur Versorgung	ja



<b>Tabelle</b>	<b>Attribut</b>	<b>Bemerkung</b>	<b>sichtbar für Anwender</b>
		der hüftgelenknahen Femurfraktur), erscheint üblicherweise im Titel des Formulars	
Bogen	bezeichnung	Bezeichnung des Teildatensatzes	ja
Bogen-Feld	gliederungAufBogen	Nummer des Datenfelds, dient bei umfangreicheren Bögen zur besseren Orientierung	(ja)
Bogen-Feld	bezeichnung	Bezeichnung des Datenfelds	ja
Bogen-Feld	ergaenzendeBezeichnung	Ergänzende Bezeichnung zum Datenfeld, kann z. B. durch Wahl der Schrift von der Bezeichnung abgesetzt werden	ja
Feld	laenge	Definiert die Länge des Eingabefelds. Für die Gestaltung des Eingabefelds sind weitere Informationen aus der Datenbank wichtig (z. B. <code>Feld.nachKommaLaenge</code> ).	ja
Feld	einheit	Einheiten (wie z. B. <code>ml</code> ) müssen angezeigt werden.	ja
BasisTyp	format	Formatanweisungen (wie z. B. <code>TT.MM.JJJJ</code> ) sollen – sofern nicht durch übergeeignete Eingabefelder unterstützt – für den Anwender angezeigt werden.	(ja)
SchlüsselWert	Code	Bei Schlüsselfeldern sollen die Codes möglichst in Auswahllisten angezeigt werden. Bei einigen Realisierungsvarianten (z. B. Checkbox) kann auf die Anzeige der Codes verzichtet werden.	(ja)
SchlüsselWert	bezeichnung	Bei Schlüsselfeldern müssen die Textdefinitionen der Codes (z. B. in einer Auswahlliste) angezeigt werden. Für die Sortierung sind die Attribute <code>sortierNrVerwendet</code> und <code>zahl</code> der Tabelle <code>Schlüssel</code> relevant.	ja
Ab-schnitt	bezeichnung	Die Überschriften sind wichtig für die Strukturierung und das Verständnis des Datensatzes und müssen deshalb in der Dokumentationssoftware angezeigt werden.	ja

Werden Datenfelder eines QS-Datensatzes aus Fremdsystemen über Schnittstellen importiert, so sollen die übernommenen Daten auch in der Erfassungssoftware angezeigt werden. Es ist für den Anwender wichtig, die vollständigen QS-Daten im Kontext eines „QS-Formulars“ zu sehen und auch auf Richtigkeit und Vollständigkeit zu prüfen.

### Allgemeine Grundsätze für die Gestaltung der Eingabemaske

Die Grundsätze für die Plausibilitätsprüfungen wirken sich insbesondere auf die Gestaltung der Benutzeroberflächen in der Erfassungssoftware aus. Durch die funktionale Gestaltung sollte ein Kompromiss zwischen Dateneingabekomfort einerseits und Zwang zur aktiven Eingabe korrekter Daten andererseits gefunden werden. Im Folgenden werden die Regeln für die Gestaltung von Benutzeroberflächen aufgeführt:<sup>1</sup>

- Keine Suggestion von Feldinhalten durch Vorbelegung (Defaults): Oberstes Prinzip bei der Gestaltung der Benutzeroberflächen ist, dass dem Anwender des Programms keine Angaben suggeriert werden. Insbesondere darf keine Vorbelegung mit Standardwerten erfolgen, die den „Nicht-Problemfall“ dokumentieren.



#### Hinweis

Eine Vorbelegung mit Standardwerten ist nicht zulässig. Die Übernahme von im KIS/AIS vorhandenen Angaben in die Dokumentation ist hingegen zulässig. Dies gilt ebenfalls für verschiedene Strukturparameter, die durch den Softwareanbieter in den Bogen übernommen werden können. Beispielsweise kann die Angabe des Datenfelds `STANDORTOPS` als Strukturparameter übernommen werden, wenn diese Angaben vorliegen.

---

- Verwendung der vorgeschriebenen Fehler- und Warnmeldungen bei feldübergreifenden Regeln: Die Fehler- und Warnmeldungen sind so formuliert, dass sie möglichst nicht suggerieren, auf welche Weise sich widersprechende Angaben korrigiert werden sollen. Insofern sollen sie wörtlich übernommen werden.
- Keine zusätzlichen Ober-/Untergrenzen für Maße, Zeitdauern und Anzahlen: Außer den durch die Datenfeldbeschreibung und die Plausibilitätsregeln vorgegebenen Wertebereichen darf in Erfassungsprogrammen keine Einengung möglicher Merkmalsausprägungen in Wertefeldern erfolgen.
- Zwang zur aktiven Entscheidung zwischen „ja“ und „nein“: An entsprechenden Stellen in den Dokumentationsbögen, bei denen die Auswahl „0“ (nein) und „1“ (ja) (vgl. z. B. Schlüssel `JN`) zu treffen ist, darf keine Voreinstellung des Wertes im Eingabefeld erfolgen. Es besteht somit der Zwang zur Eingabe eines Wertes. Nur an Stellen, an denen im Erfassungsformular lediglich „1“ als Option angegeben wird, soll die Nicht-Eingabe eines Wertes als Verneinung interpretiert werden. Hintergrund dieser Differenzierung ist, dass einerseits in qualitätskritischen Bereichen eine Unterscheidung zwischen „keine Angabe“ und „nein“ erfolgen muss, es andererseits

---

<sup>1</sup> Externe Systeme, die Daten an ein Erfassungsprogramm übergeben, sollten diese Grundsätze sinngemäß anwenden.

der Benutzerakzeptanz abträglich ist, wenn diese Systematik auch an allen anderen Stellen durchgängig verfolgt wird.

- Umrechnung von Einheiten bei numerischen Feldern: In Einzelfällen ist es aus Anwendersicht hilfreich, wenn die Eingabemaske die Dokumentation von Messwerten in Einheiten ermöglicht, die von den spezifizierten Einheiten abweichen. Diese Funktionalität sollte möglichst in die Erfassungssoftware integriert werden, um den Dokumentationsaufwand zu verringern.
- Gestaltung von Eingabemasken mit Layout-Feldgruppen: Sogenannte Layout-Feldgruppen sollen in der Erfassungsmaske separat kenntlich gemacht werden. Hierbei handelt es sich um Datenfelder, die zu einer logischen Gruppe zusammengefasst werden können. Feldgruppen können Filterfelder oder abhängige Felder beinhalten. Abhängige Felder von Layout-Feldgruppen werden auf den generierten Dokumentationsbögen ausgegraut dargestellt. Hierbei handelt es sich um Feldgruppen, bei denen das Attribut `grauWennNegativ` in der Datenbanktabelle `FeldGruppe` gesetzt ist. Diese sollen für die Gestaltung von Eingabemasken verwendet werden. Die Eigenschaften von Layout-Feldgruppen sind in Abschnitt 9.3.8 erläutert.
- Empfehlung zur Umsetzung von Layout-Feldgruppen: Ist das Attribut `grauWennNegativ` gesetzt, so darf die Benutzereingabe für die abhängigen Felder durch die Erfassungssoftware deaktiviert werden, falls die negative Filterbedingung zutrifft. Bei der Umsetzung muss Folgendes sichergestellt werden:
  - Nach jeder Änderung der Inhalte der Filterfelder im Erfassungsformular muss das Programm die Filterbedingung der Feldgruppe evaluieren und ggf. eine Aktualisierung der Oberfläche durchführen.
  - Die Benutzereingabe für die abhängigen Felder darf nur dann deaktiviert werden, wenn keines dieser Felder ausgefüllt ist. Ansonsten ist der Anwender auf eine Plausibilitätsverletzung hinzuweisen.
  - Wenn nach einer Benutzereingabe die positive Filterbedingung zutrifft, so sind ggf. vorher deaktivierte Eingabefelder wieder zu aktivieren.
  - Deaktivierte Felder dürfen nicht ausgeblendet werden.

## 3.2 Einrichtungsidentifizierende Daten

In Hinblick auf eine standortbezogene Auswertung und Berichterstattung sind einrichtungsidentifizierende Daten zu dokumentieren.

### Einrichtungsidentifizierende Daten des Krankenhauses

Wichtigstes Merkmal zur Identifikation der Einrichtung ist die Angabe des behandelnden Standorts. Die Standortnummern sind ab der Spezifikation 2020 neunstellig und bundesweit eindeutig. Sie werden gemäß § 293 Abs. 6 SGB V Vereinbarung über ein bundesweites Verzeichnis der Standorte der nach § 108 SGB V zugelassenen Krankenhäuser und ihrer Ambulanzen vergeben.

### 3.3 QS-Daten

Es sind alle erforderlichen Daten zu dokumentieren. Hierbei kann eine automatische Übertragung der QS-Daten aus dem KIS/AIS möglich sein.

Der Softwareanbieter muss (z. B. bei Änderungen im Nachhinein) sicherstellen, dass Aktualisierungen im KIS/AIS zwischen der Anwendungssoftware und der Dokumentationssoftware kommuniziert werden.

Grundsätzlich ist hier eine Übereinstimmung aller relevanten Angaben sicherzustellen.



#### Achtung

Daten im Bogen müssen regelmäßig aktualisiert werden, um mit dem aktuellen Stand der Daten im KIS/AIS übereinzustimmen.

Eine automatisierte Aktualisierung von Datenfeldern im Dokumentationsbogen kann ein Softwareanbieter nur durchführen, wenn es sich um Daten handelt, die im KIS/AIS vorliegen. Hierbei handelt es sich in der Regel um Daten, die auch im Rahmen der Technischen Anlagen nach § 301 SGB V und § 21 KHEntgG definiert sind. Die Aktualisierung manuell zu befüllender Datenfelder liegt in der Verantwortung des Leistungserbringers.

---

### 3.4 Plausibilitätsprüfungen

Fehlende und widersprüchliche Angaben in den Datensätzen sollen durch umfangreiche Plausibilitätsprüfungen verhindert werden. In der Dokumentationssoftware muss die vollständige Plausibilitätsprüfung für jeden Datensatz spätestens bei Dokumentationsabschluss erfolgen. Teile der Plausibilitätsprüfungen sollen bereits während der Erfassung erfolgen. Dadurch wird sichergestellt, dass ein aufwendiges Korrekturverfahren – Verschlüsselung und Übermittlung der Datensätze, Prüfung, Fehlerprotokollierung (Datenflussprotokoll), Korrektur der Dokumentation und erneute Übermittlung des Datensatzes – weitgehend entfällt.

Die Datenannahmestellen führen für jeden Datensatz alle harten Plausibilitätsprüfungen der Spezifikation durch. Bei einer Regelverletzung ist der Datensatz zurückzuweisen. Die Datenannahmestelle darf keine zusätzlichen (in der Spezifikation nicht definierten) Plausibilitätsprüfungen durchführen.

Es gelten folgende Grundsätze für die Plausibilitätsprüfungen:

- Alle Felder müssen ausgefüllt sein, wenn keine anderen logischen Sachverhalte dem entgegenstehen.
- Jedes Feld, das auszufüllen ist, muss einen sinnvollen Feldinhalt haben.
- Es wird jede harte Plausibilitätsprüfung vorgenommen, die definiert ist.
- Harte Plausibilitätsprüfungen werden nur vorgenommen, wenn Sachverhalte zwingend miteinander gekoppelt sind.
- Es werden keine Sachverhalte suggeriert (keine Default-Werte, keine Vorbelegungen, keine Profile und/oder Fehlermeldungen werden vorgegeben).

- Keine Angabe (bzw. kein Feldinhalt) wird ergänzt oder gelöscht.

### Arten der Plausibilitätsprüfungen

Es wird zwischen zwei Arten von Plausibilitätsprüfungen (QSDOK.Regeln) unterschieden:<sup>2</sup>

- Harte Regeln
- Weiche Regeln in der Dokumentationssoftware

Die zwei Arten der Regeln werden in unterschiedlichen Kontexten (Software bzw. Datenentgegennahme) durchgeführt und haben unterschiedliche Konsequenzen. Tabelle 2 gibt einen Überblick:

Tabelle 2: Arten der Plausibilitätsprüfungen

Art der Prüfung	Kürzel	Prüfung durch Software	Prüfung durch Datenannahmestelle	Konsequenz: Verhindert Dokumentationsabschluss/Datenentgegennahme
Hart	H	Ja	Ja	Ja
Weich	W	Ja	Optional	Nein

### Harte Plausibilitätsprüfungen (QSDOK.Regeln)

Harte Prüfungen sind sowohl in der Dokumentationssoftware als auch bei der Datenentgegennahme anzuwenden. Bei einer harten Regelverletzung ist:

- Ein Dokumentationsabschluss eines Vorgangs unzulässig.
- Ein Datensatz von der entgegennehmenden Stelle zurückzuweisen.

Die in der Technischen Dokumentation und der Datenbank definierten Plausibilitätsprüfungen sind hart, außer wenn sie explizit als weich gekennzeichnet sind.

### Weiche Plausibilitätsprüfungen (QSDOK.Regeln)

Die weichen Plausibilitätsprüfungen der Spezifikation sind von der Dokumentationssoftware bis spätestens zum Dokumentationsabschluss durchzuführen. Bei einer Regelverletzung erhält der Benutzer einen Warnhinweis, anhand dessen er entscheidet, ob eine Änderung von Feldinhalten notwendig ist. Ebenso wie harte Plausibilitätsprüfungen müssen weiche Regeln immer vom Softwareanbieter umgesetzt werden.

### Einzelregeln

Einzelregeln sind in der QS Dokumentation Datenbank in Regelsyntax in der Tabelle `Regeln` hinterlegt. Außerdem gibt es Regeln, die nur in Form von Feldeigenschaften – nicht aber in Regelsyntax – in der Datenbank hinterlegt sind. Die standardisierten Fehlertexte dieser Prüfungen sind Abschnitt 9.3.9 zu entnehmen.

<sup>2</sup> Die Komponentensicht ist in Abschnitt B „Plausibilitätsprüfungen“ beschrieben.

Die Dokumentationssoftware muss sowohl die harten als auch die weichen feldbezogenen Prüfungen ausführen. Die Evaluation soll direkt bei der Dateneingabe geschehen. Fehleingaben sollen dem Benutzer direkt mitgeteilt werden. Einige Prüfungen erübrigen sich durch adäquate Gestaltung von Eingabemasken, z. B. durch Bereitstellung von Auswahlmenüs für Schlüsselkodes. Bei Regelverletzung muss die Software dem Benutzer verständliche Fehlertexte anzeigen.

Bei der Datenentgegennahme sind alle harten Prüfungen zu evaluieren und bei Regelverletzung die unten definierten standardisierten Fehlertexte im Rahmen des im Abschnitt 9.3.9 beschriebenen Korrekturverfahrens an die Einrichtung zu übermitteln.

#### *Feldgruppenregeln*

Datenfelder (Bogenfelder) eines Moduls können zu einer Feldgruppe zusammengefasst werden, um logische Abhängigkeiten von Bogenfeldern abzubilden. Das bedeutet in der Praxis, dass der Anwender daran gehindert wird, Felder mit Werten auszufüllen, die der Logik der Feldgruppe widersprechen.

Die explizite Definition von Feldgruppen strukturiert sowohl die Bogenfelder als auch die Plausibilitätsregeln, indem diese die Bogenfelder eines Moduls zu einer logisch zusammenhängenden Gruppe von Feldern zusammenfassen. Die Feldgruppen ergeben sich dabei indirekt aus der Definition von Plausibilitätsregeln.

Die Abbildung von Feldgruppenregeln in der Datenbank ist in Abschnitt 9.3.8 erläutert.

## 4 Exportdaten

Die Exportdateien werden beim Leistungserbringer erstellt und an die Datenannahmestelle weitergeleitet. Der Datenaustausch erfolgt im XML-Format.

### 4.1 Erzeugen der Exportdatei

Die zu exportierenden Daten werden vom Dokumentationssystem in Exportdateien geschrieben und die entsprechenden Vorgänge (jeweils identifiziert durch eine `VorgangsnrGuid`) im absenden- den Dokumentationssystem als „exportiert“ markiert.

In den nachfolgenden Abschnitten wird der Inhalt der Exportdateien beschrieben:

#### Export von Teildatensätzen

Beim Export einer Dokumentation durch ein Dokumentationssystem werden die Inhalte der für den betreffenden Vorgang angelegten Teildatensätze hierarchisch in den XML-Code des passenden Basisbogens geschrieben und können nur gemeinsam mit dem Inhalt des Basisbogens exportiert werden. Die Struktur der Einbettung ist durch den Datentyp des Exportmoduls im Schema definiert.

Die Reihenfolge der Unterbögen (Teildatensätze) ist in der Spezifikationsdatenbank festgelegt. In der Tabelle `Bogen` ist in der Spalte `SortierNr` die Reihenfolge der Unterbögen bei der Erfassung und beim Export aufgeführt.

#### Aufbau der Exportdatei

Die innere Struktur der Exportdatei ergibt sich aus der Datenfeldbeschreibung der einzelnen Module. Aufbauend auf dieser Beschreibung wird ein XML-Schema abgeleitet. Die Struktur der Exportdatei wird durch entsprechende XML-Schemata festgelegt (Abschnitt 10).

Das XML-Schema beschreibt und definiert die Struktur des XML-Dokuments (Exportdatei) sowie den Inhaltstyp (Datentypen der einzelnen Bögen und Felder).

Die Exportdateien sind wie folgt aufgebaut:

- Header-Bereich enthält die Metadaten (administrative Daten)
- Body-Bereich enthält die tatsächlichen Daten der Datenlieferung

Das Encoding UTF-8 (ohne BOM) muss in der XML-Deklaration angegeben werden:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Die Eingrenzung auf "LATEINISCHE ZEICHEN IN UNICODE" ist zu beachten, sie kann aber in der XML-Deklaration nicht kenntlich gemacht werden. Die Eingrenzung auf "LATEINISCHE ZEICHEN IN UNICODE" wurde in den Schemata umgesetzt durch den Typ `String.Latin` aus dem Namespace `xoev-lc`.

## Sonderzeichen in XML

Die folgenden Zeichen müssen in den Zeichendaten (Werten) der Exportdatei „maskiert“ werden:

- < mit &lt;
- > mit &gt;
- & mit &amp;
- " mit &quot;
- ' mit &apos;

Dies ist insbesondere beim Export der Freitextfelder zu beachten. Jedes dieser Zeichen unmaskiert (bzw. nicht als Entity referenziert) würde die Wohlgeformtheit einer XML-Datei verletzen und zu einer Ablehnung der Datenannahme beim Datenempfänger führen.

## Exportfelder eines Exportmoduls

Einen Überblick über die zu exportierenden Felder eines Exportmoduls liefert die Tabelle `ExportFormatExportModul`.

## Export von Listenfeldern

Alle Elemente von Listenfeldern werden exportiert, ohne die Nummer des Listenfelds (im Namen des Exportfelds) an den Namen des Listenfelds anzuhängen.

---

### Beispiel:

XML-Darstellung eines Listenfeldes

```
<MONATE V="3" />
```

```
<MONATE V="2" />
```

```
<MONATE V="4" />
```

---

## Zusatzfelder des Datenexports

Zusatzfelder und administrative Felder im Header, die nicht in der Datenfeldbeschreibung (Tabelle `BogenFeld`) eines Moduls enthalten sind, sind von der Dokumentationssoftware zu füllen.<sup>3</sup>

Die Zusatzfelder sind in der Tabelle `ZusatzFeld` definiert. Das übertragene Speicherdatum `DokAbschlDat` (Datum des Dokumentationsabschlusses bzw. der Freigabe des Datensatzes für den Export) ist nicht Teil der Datenbank für Auswertungen und wird nur für organisatorische Zwecke verwendet.

Bei der Organisation im XML-Dokument ist weiter die Abfrage `vExportZieleXml` aus den administrativen Objekten zu berücksichtigen (Abschnitt 9.7). Die Informationen in dieser Abfrage

---

<sup>3</sup> Hier gilt demnach nicht der Grundsatz, dass Felder nicht vorbelegt sein dürfen.



schließen aus, dass bestimmte Felder dem Element `<qs_data>` zugeordnet werden, indem anhand eines XPath-Ausdrucks (`xmlXPath`) auf die richtige Stelle im XML-Dokument verwiesen wird.

**Organisation der Exportfelder im XML-Dokument**

Die Exportfelder sind abhängig von der Modul- und Teildatensatzzugehörigkeit des Datensatzes im Element `<qs_data>`, einem Unterelement des Elements `<case>` unterzubringen.

**Stornierung**

Stornierungen unterscheiden sich in der XML-Struktur nicht von anderen Exportdateien mit `create` oder `update` im XML-Attribut `case/case_admin/action/@v`. Soll ein Datensatz beim Empfänger storniert werden, dann muss `action/@v` dem Wert `delete` entsprechen. Das XML-Element `qs_data` ist bei Stornierungen optional.

**4.2 Datenprüfung**

Zusätzlich zur bereits im Rahmen der Erfassung durchzuführenden feldbezogenen und -übergreifenden Plausibilitätsprüfungen muss nach dem Datenexport die gesamte Struktur der XML-Datei durch aus der Spezifikationsdatenbank abgeleitete Schema geprüft werden.

Welches Schema für einen Leistungserbringer vorgesehen ist, zeigt Tabelle 3.

Tabelle 3: XML-Schemata zur Prüfung vor der Verschlüsselung am Beispiel von QSFFx

Richtlinie	Bereich	Schema
QSFFx	Krankenhaus (Datenfluss IQTIG)	LE_und_BAS_data_flow_LE_BAS.xsd
QSFFx	Krankenhaus (Datenfluss LVKK/EK)	LE_und_LVKK_data_flow_LE_LVKK.xsd

Die einfachste Lösung für die Prüfung der Datenstruktur und der Plausibilität liegt in der Nutzung eines Datenprüfprogramms, das auf der Basis von XSLT die aus der Spezifikationsdatenbank QSDOK ausgeleiteten Plausibilitätsregeln in dem XML-Dokument prüft und Verstöße entsprechend im XML-Code dokumentiert. Ein solches Datenprüfprogramm setzt einen Standard für die Güte der Daten, der unbedingt einzuhalten ist. Die Datenprüfung muss an der Exportdatei vor der nachfolgend beschriebenen XML-Verschlüsselung vorgenommen werden.

Das Datenprüfprogramm ersetzt jedoch nicht die Verpflichtung der Softwareanbieter, schon bei der Eingabe der Daten eines Datensatzes, d.h. dokumentationsbegleitend, für die Einhaltung der Plausibilitätsregeln zu sorgen.

Details zur Verwendung können der Dokumentation des Datenprüfprogramms entnommen werden (Abschnitt 11.1).

### 4.3 Beispiele für Exportdateien

Der Ordner XML-Beispiele in der Spezifikationskomponente XML-Schema enthält für jedes Exportmodul eine valide Beispieldatei.

### 4.4 Umgang mit Umlaut-Domains und E-Mail-Adressen

Die Nutzung von Umlauten ist durch die Internationalized Domain Names (IDN) möglich. Domainnamen wie z. B. <http://www.qualitätsklinikverbund.de/> sind damit nutzbar. Da die Verwendung von Zeichen aus einem Unicode-Zeichensatz in Domainnamen und E-Mail-Adressen Probleme in Anwendungsprogrammen bereiten, wurde das ASCII Compatible Encoding (ACE) entwickelt, in dem Regeln zur Überführung von Unicode-Zeichen in ASCII-Zeichen festgelegt wurden. Zu jeder Umlautdomain (IDN-Domain) wird parallel immer eine ACE-Domain registriert. Für das oben genannte Beispiel <http://www.qualitätsklinikverbund.de/> würde der ACE-Name `xn--qualittsklinikverbund-b2b.de` lauten.

E-Mail-Adressen dürfen im lokalen Teil (vor dem @-Zeichen) keine Umlaute enthalten. Die E-Mail-Adresse `müller@muster.de` wäre nicht möglich. Möglich ist lediglich `mueller@muster.de`. Es existieren experimentelle Request for Comments (RFCs)<sup>4</sup>, in denen UTF-8-codierte, internationale E-Mail-Adressen ermöglicht werden. Dies wird jedoch nicht flächendeckend im Produktiveinsatz genutzt, aus diesem Grund wird in dieser Spezifikation von der Annahme ausgegangen, dass der lokale Teil der E-Mail-Adresse immer durch ASCII-Zeichen repräsentiert wird.

In den E-Mail-Elementen des XML-Schemas (siehe Abbildung 5) sowie in der Auftragsdatei (siehe Abschnitt 5.1) sind ausschließlich Zeichen des ASCII-Zeichensatzes erlaubt. E-Mail-Adressen mit einem IDN Domainnamen sind entsprechend den ACE-Regeln durch ASCII-Zeichen darzustellen und zu übermitteln.

## 5 Datenübermittlung

Die Datenübermittlung im Rahmen der Strukturabfragen erfolgt zwischen dem Leistungserbringer als Datenquelle und einem oder mehreren der im Folgenden aufgeführten Institutionen als Datenempfänger:

- Landesverbände der Krankenkassen oder Ersatzkassen
- Institut für Qualitätssicherung und Transparenz im Gesundheitswesen und die
- Landesaufsichtsbehörden

In Abbildung 1 sind die Übertragungswege und die Paragraphen der PPP-RL, der QFR-RL und der QSFFX-RL, in denen die entsprechende Kommunikation geregelt ist, schematisch dargestellt.

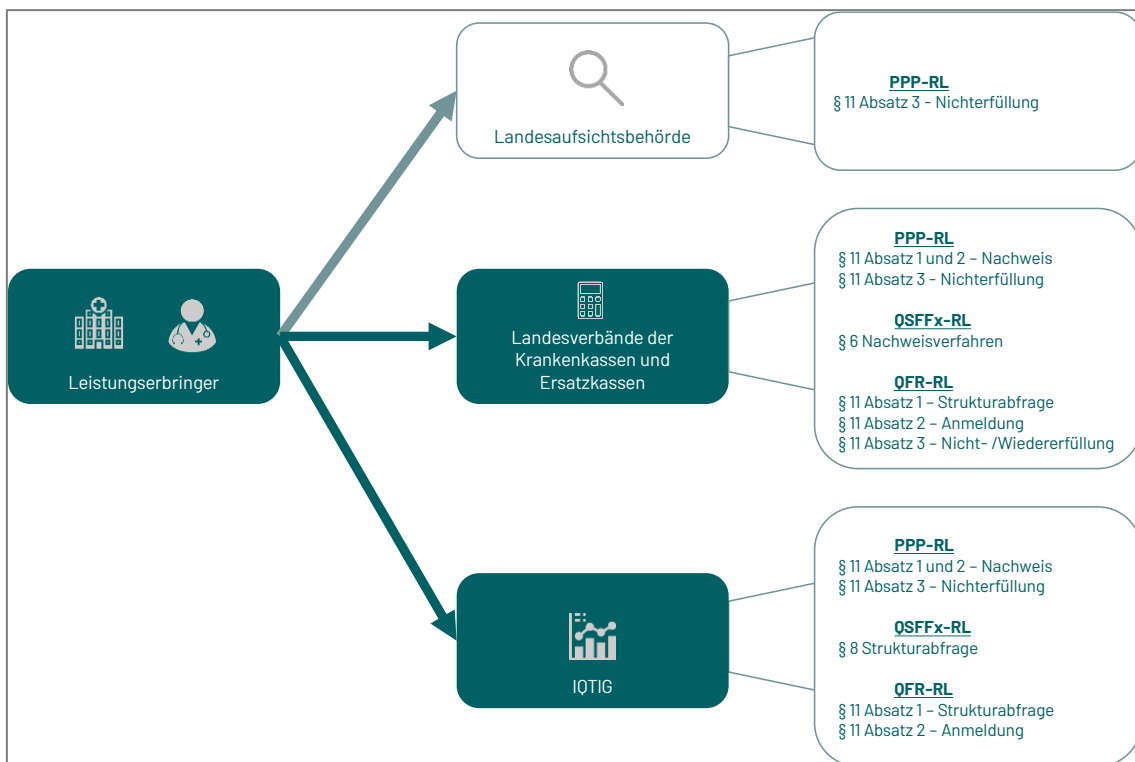


Abbildung 1: Kommunikationsbeziehungen zwischen Leistungserbringer, Landesaufsichtsbehörden, Landesverbänden der Krankenkassen und Ersatzkassen sowie dem IQTIG inkl. Richtlinienvorgaben

Neben den in Abbildung 1 skizzierten Kommunikationsbeziehungen fordert die QSFFX-RL in § 8 eine Mitteilung des IQTIG über die Nichterfüllung von Dokumentationspflichten an die Landesverbände der Krankenkassen und Ersatzkassen.

Aufgrund der Heterogenität bei den technischen Gegebenheiten der in den Kommunikationsprozess eingebundenen Teilnehmern wird im Rahmen dieser Spezifikation eine einheitliche Lösung

angestrebt, die bei der Datenaufbereitung und dem Datenexport beim Leistungserbringer möglichst einheitlich und ressourcenschonend umgesetzt werden kann und auch die unterschiedlichen Vorgaben bei den Datenempfängern berücksichtigt.

Die Kommunikation zwischen dem Leistungserbringer und dem IQTIG erfolgt auf Grundlage der durch die verpflichtende Qualitätssicherung etablierten Prozesse und Strukturen. Die Datenstrukturen (XML-Dateien) werden entsprechend der etablierten Vorgaben spezifiziert und über das E-Mail-Verfahren der Qualitätssicherung übermittelt. Bezüglich der Registrierung der Teilnehmer und der Verschlüsselung der zu übermittelnden Daten erfolgt im Rahmen der Strukturabfragen eine Weiterentwicklung der bestehenden Technologien. In der Qualitätssicherung müssen sich die Verfahrensteilnehmer über den Verfahrenssupport registrieren und erhalten mit der Registrierung ein Passwort, um die zu übermittelnden Daten zu verschlüsseln. Mit den Strukturabfragen wird eine Registrierung entfallen und die Verschlüsselung der Daten unter Nutzung des öffentlichen Schlüssels des IQTIG bzw. der LAB erfolgen. Nähere Informationen werden in den entsprechenden Abschnitten zur Registrierung, Verschlüsselung und dem Datenversand gegeben.

Die Kommunikation zwischen dem Leistungserbringer und den Landesverbänden der Krankenkassen und der Ersatzkassen ist durch die Gemeinsamen Grundsätze Technik (GGT) für die elektronische Datenübermittlung gemäß § 95 SGB IV und den technischen Anlagen geregelt. Für die Datenübermittlung im Rahmen der Strukturabfragen wird das E-Mail-Verfahren gemäß GGT genutzt. Die sogenannten Nutzdaten werden in der gleichen XML-Struktur vom Leistungserbringer zu den Landesverbänden der Krankenkassen und Ersatzkassen übermittelt, die auch für die Übermittlung an das IQTIG genutzt wird. Die Verschlüsselung und Signatur erfolgt gemäß den Vorgaben der GGT. Die detaillierte Umsetzung wird in den folgenden Abschnitten erläutert.

## 5.1 Allgemeine Vorgaben

In diesem Abschnitt werden allgemeine Regelungen in Bezug auf die Datenübertragung beschrieben.

## 5.2 Eindeutige Kennzeichnung der XML-Exportdateien

Jede Exportdatei wird durch einen Globally Unique Identifier (GUID) von der Software gekennzeichnet. Eine (GUID) ist eine weltweit eindeutige Zeichenfolge mit 128 Bit, die eine Implementierung des Universally Unique Identifier Standards (UUID) darstellt.

GUIDs haben das Format XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX, wobei jedes X für ein Zeichen aus dem Hexadezimalsystem steht und damit eine Ziffer 0–9 oder ein Buchstabe A–F sein kann.

### Erläuterung zur GUID:

- Die GUID wird im Exportprozess von der Software einer bestimmten Exportdatei zugewiesen.

- Das Dokumentationssystem ordnet jeder an eine Datenannahmestelle übermittelten Exportdatei eine eindeutige GUID zu.
- Diese GUID wird im XML-Code des Dokuments als ID gesetzt. Sie muss bei dateibasierten Übertragungsverfahren in der Dateibenennung verwendet werden.
- Eine GUID wird durch eine gelungene Transaktion zwischen den jeweils beteiligten Übertragungspartnern verbraucht. Jede Datenannahmestelle muss dafür sorgen, dass ein eindeutig über die GUID identifizierbares Dokument nur einmal denselben Verarbeitungsschritt durchlaufen kann. Anderenfalls ist die Verarbeitung mit einem entsprechenden Fehlerprotokoll abzulehnen.

### 5.2.1 Identifizierung von Datensätzen anhand von Vorgangsnummern

Die Vorgangsnummer (auch Datensatz-ID oder ID genannt) kennzeichnet in eindeutiger Weise jeden dokumentierten Vorgang eines Dokumentationssystems®.

Im einfachsten Fall könnten die Vorgangsnummern jeweils um 1 inkrementiert werden, wenn ein neuer Datensatz angelegt wird. Wenn zwei QS-Dokumentationen angelegt werden, so müssen auch unterschiedliche Vorgangsnummern vergeben werden. Die Vorgangsnummer ist daher allein ein Merkmal des QS-Dokumentationssystems, um einen Datensatz eindeutig identifizieren zu können. Eine Vorgangsnummer darf keine Rückschlüsse auf Personen ermöglichen.

Die QS-Dokumentationssoftware verwaltet jahresübergreifend die Vorgangsnummern der QS-Dokumentationen. Sie soll dem Leistungserbringer eine Zuordnung der Vorgangsnummern zu seiner QS-Dokumentation ermöglichen.

Zusätzlich gibt es einen 36-stelligen pseudozufälligen GUID, der die Datensätze in der jeweiligen Datenannahmestelle bundesweit eindeutig identifiziert.

#### **Annahme oder Ablehnung unterschiedlicher Versionen eines Datensatzes**

In der jeweiligen Datenannahmestelle – und nur in dieser – können Vorgänge anhand der VorgangsnrGuid identifiziert werden. Eine eindeutige und dauerhafte Identifizierung eines Vorganges im gesamten Datenfluss ist in der Datenannahmestelle nur mit der VorgangsnrGuid möglich. In verschiedenen Datenannahmestellen liegen verschiedene VorgangsnrGUID vor.

Bei den Leistungserbringern muss gelten, dass Datensätze mit identischer Vorgangsnummer immer auf dieselbe VorgangsnrGuid verweisen. Diese Beziehung soll eineindeutig sein, das heißt, Datensätze mit identischer VorgangsnrGuid müssen umgekehrt immer auf dieselbe Vorgangsnummer verweisen.

Der für einen bestimmten Vorgang gespeicherte Datensatz kann nur durch eine neuere Version (mit höherer Versionsnummer) überschrieben werden.<sup>4</sup> Unterschiedliche Versionen eines Datensatzes müssen demselben PrimärModul<sup>5</sup> zugeordnet sein. Ein Datensatz mit einer Vorgangsnummer, die bereits unter einem anderen Modul eingeschickt wurde, wird abgelehnt.

### 5.2.2 Benennung der Exportdateien

Die Daten werden als XML-Datei aus dem Dokumentationssystem exportiert. Die Exportdatei muss nach dem folgenden Schema benannt werden:

Syntax: <GUID>\_<Inhaltskennung><Protokolltyp>\_<Rolle Absender>.xml

Tabelle 4: Benennungselemente der Exportdateien

Element	Bedeutung
GUID	Die verwendete GUID ist die im Dokument verwendete ID des Datenpakets.
Inhaltskennung	Q → QS-Daten
Protokolltyp	T → Transaktionsprotokoll bzw. Empfangsbestätigung D → Datenflussprotokoll
Rolle Absender	LE → Leistungserbringer DAS → Datenannahmestelle

#### Beispiele:

47d16341-9e27-4e75-a27e-b791fbbd2dc8\_Q\_LE.xml

(QS-Daten eines Leistungserbringers)

47d16341-9e27-4e75-a27e-b791fbbd2dc8\_QD\_DAS.xml

(QS-Daten-Datenflussprotokoll einer DAS)

In den Verfahren der Strukturabfrage nimmt das IQTIG (BAS) gegenüber den Leistungserbringern die Rolle einer Datenannahmestelle ein. Die Datenflussprotokolle werden mit dem Suffix \_QD\_DAS geschickt.

### Ausgangvalidierung gegen das XML-Übertragungsschema

Bevor das Dokument an den Empfänger weitergeleitet wird, muss die XML-Datei gegen das Übertragungsschema auf Gültigkeit geprüft werden.

Die Vorteile der Ausgangvalidierung sind:

- Sicherstellung der Datenintegrität nach Verarbeitung der Daten

<sup>4</sup> Gegebenenfalls ist der geänderte Datensatz mit einer neuen Versionsnummer zu übermitteln.

<sup>5</sup> Jeder Datensatz ist einem Primärmodul zugeordnet.

- Frühe Feststellung von Fehlerquellen in der eigenen Datenverarbeitung
- Entlastung der nachfolgenden Datenservices von nicht validen Daten
- Vermeidung des Versands von Daten, die gegen den Datenschutz verstoßen

Das an einer Übertragungsstelle gültige Schema, kann der Dokumentation über die Schemafamilie entnommen werden.

---

**Beispiel:**

Der Leistungserbringer verwendet bei der QSFFx Richtlinie das Schema zur Schnittstelle LE-/BAS:

```
interface_LE_BAS\interface_LE_BAS.xsd
```

---

Die Validierung kann über zahlreiche frei verfügbare Tools erfolgen.<sup>6</sup> Für diese Validierung kann auch das Datenprüfprogramm des IQTIG verwendet werden.

### **Erzeugung der Transaktionsdatei für die Übertragung via E-Mail nach den Vorgaben der QS**

Für die Übermittlung der Daten über nicht gesicherte Übertragungswege, wie z. B. E-Mail an das IQTIG als Bundesauswertungsstelle oder die Landesaufsichtsbehörden gemäß PPP-RL, muss die Exportdatei in eine Transaktionsdatei umgewandelt werden. Hierfür findet das symmetrische Verschlüsselungsverfahren Advanced Encryption Standard (AES) in Verbindung mit einer asymmetrischen Verschlüsselung basierend auf Zertifikaten Anwendung.

Das IQTIG stellt ein Verschlüsselungsprogramm bereit, mit dem eine verfahrenskonforme Transportverschlüsselung durchgeführt werden kann. Das Einbinden der Funktionen des Verschlüsselungsprogramms erfolgt über einen Befehlszeilenaufwurf mit Parametern oder über die grafische Benutzeroberfläche. Das Verschlüsselungsprogramm übernimmt auch die Dateibenennung der Transportdatei mithilfe von übergebenen Parametern.

Die Leistungserbringer nutzt zur Verschlüsselung die öffentlichen Schlüssel des Datenempfängers (IQTIG/LAB). Das Verschlüsselungsprogramm erstellt die Exportdatei unter Nutzung der Zertifikate. Die Transaktionsdatei wird wie folgt benannt:

```
T-<Zeitstempel im Format YYYY_MM_tt_hhmmss>[_<drei weitere numerische Stellen>].zip.aes
```

```
T-2021_01_10_160945.zip.aes
```

```
T-2021_01_10_114113_045.zip.aes (millisekundengenau)
```

Die drei weiteren numerischen Stellen sind optional und stellen im Prinzip Millisekunden dar. Sie sind aber nur als Unterscheidungsmerkmal notwendig, wenn innerhalb einer Sekunde mehr als

---

<sup>6</sup> <http://www.w3.org/XML/Schema>.

eine Transaktionsdatei erstellt werden soll. Wenn diese Option angewendet wird, sollen alle drei Stellen gesetzt und ggf. mit „0“ aufgefüllt sein.

### Erzeugung der Antwortdatei für die Übertragung via E-Mail

Das Datenflussprotokoll wird vom Datenempfänger nach dem gleichen Verfahren wie die Transaktionsdatei erstellt.

Im Datenfluss LE\_BAS muss die XML-Exportdatei das Attribut `/root/header/dfp_key/@v` enthalten. Der Wert ist das Passwort mit dem die Bundesauswertungsstelle das zugehörige Datenflussprotokoll symmetrisch verschlüsselt. Der Wert von `dfp_key/@v` muss durch die QS Software abrufbar sein, damit das Datenflussprotokoll auf dem Rückweg beim Leistungserbringer entschlüsselt werden kann. Die Entschlüsselung erfolgt durch das Programm Tpacker.

Schemaverletzungen wie etwa ein fehlendes Attribut `dfp_key/@v` werden durch die BAS in ein Miniprotokoll geschrieben. Das Miniprotokoll wird im Unterschied zu dem vollständigen Datenflussprotokoll unverschlüsselt an den Leistungserbringer übermittelt.

Die Bundesauswertungsstelle entfernt im Datenflussprotokoll das Element `dfp_key`.

Der verwendete Zeitstempel entspricht beim Datenflussprotokoll dem Zeitstempel der Datei, mit der die Daten an die Datenannahmestelle versandt wurden. Damit ist eine einfache, eindeutige Zuordnung zur Transaktion möglich.

---

#### Beispiel:

Die Benennung der Antwortdatei

Antwortdatei:                      A-2021\_01\_10\_094051.zip.aes  
auf die Transaktionsdatei:    T-2021\_01\_10\_094051.zip.aes

---

### Identifizierung und Authentifizierung des Einsenders

Der vorliegende Abschnitt beschreibt die notwendigen Prüfungen beim Versand per E-Mail. In diesen Fällen sind die Dateien für den Transport durch eine Transportverschlüsselung abzuschichern. Diese gesicherten „Pakete“ tragen die Datei-Endung: `.zip.aes`.

Wenn eine Datenlieferung erfolgt, sollte das entsprechende Datenpaket vor dem Öffnen zunächst überprüft und unter Umständen abgelehnt werden.

### Prüfungen des Anhangs

Für einen gültigen Datenversand muss eine `.zip.aes`-Datei der E-Mail angehängt sein, die der im Anhang näher beschriebenen Namenskonvention entspricht. Der Anhang wird mit dem privaten Schlüssel des Datenempfängers mit dem Verschlüsselungsprogramm entschlüsselt.



### 5.2.3 Abgrenzung von Test- und Regelbetrieb

Im Folgenden wird beschrieben, welche Testmöglichkeiten es im Datenfluss gibt, wie diese genutzt werden können und sollen und wie sich diese voneinander und vom Produktivbetrieb abgrenzen lassen. Die im Folgenden dargestellten Strukturen und Vorgaben werden bereits bei den Verfahrensteilnehmern der datenstützten und einrichtungsübergreifenden Qualitätssicherung des G-BA derart implementiert. Es wird empfohlen, dass diese Strukturen auch bei den LVKK/EK und bei den LAB implementiert werden.

#### Definition Test- und Echtdaten

Es wird zwischen Test- und Echtdaten unterschieden (siehe Abbildung 2).

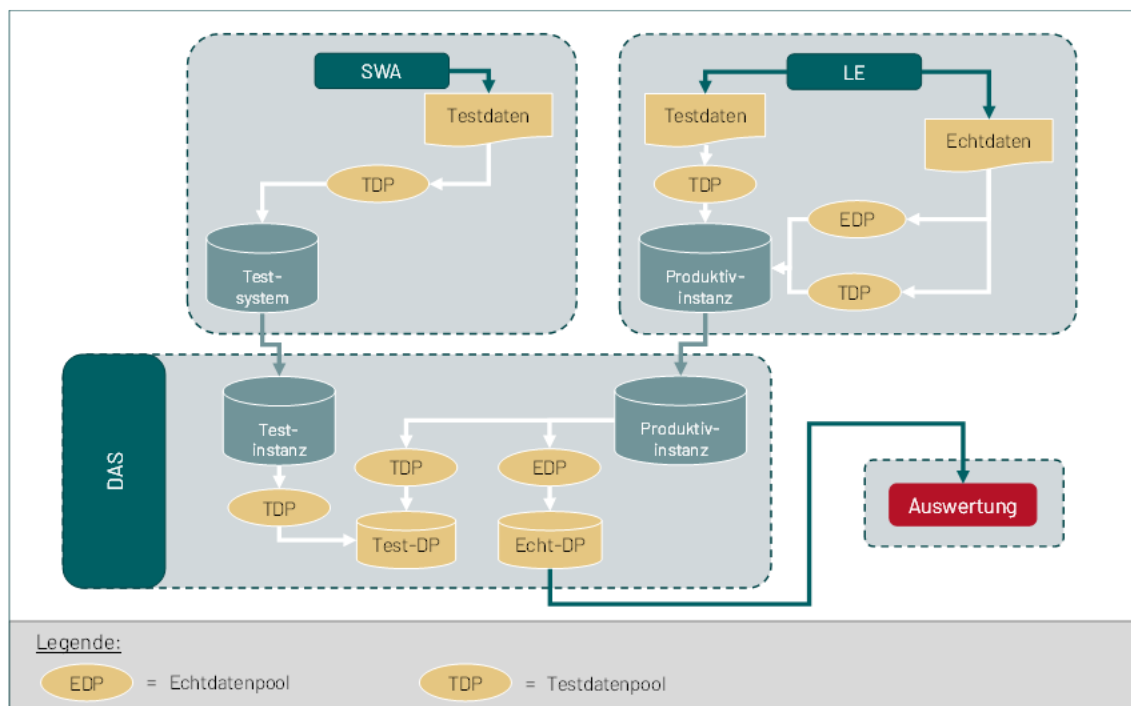


Abbildung 2: Datenflüsse im Test- und Regelbetrieb

Als **Testdaten** werden QS-Daten bezeichnet, die der Datenstruktur der Spezifikation, aber nicht Datensätzen von realen Leistungserbringern entsprechen.

Als **Echtdaten** werden solche QS-Daten bezeichnet, die der Datenstruktur der Spezifikation und Datensätzen von realen Leistungserbringern entsprechen, sodass die besonderen Regeln des Datenschutzes auf sie zutreffen.

#### Datenziele

Alle auswertenden Einrichtungen (DAS, BAS) stellen zwei Datenziele zur Verfügung. Diese Datenziele werden als

- Testdatenpool bzw.

- Echtdatenpool

bezeichnet.

Die Datensicherheit bemisst sich in allen Datenpools an den Maßstäben, die für QS-Daten gelten. Das bedeutet, dass prinzipiell alle zwei Datenziele geeignet sind Echtdaten entgegenzunehmen, weil die Sicherheit gewährleistet wird.

Das Datenziel wird im Headerbereich des XML-Dokuments festgelegt (/document/data\_target).

### **Testdatenpool**

Der Testdatenpool ist sowohl für generierte Testdaten als auch für Echtdaten, die zu Testzwecken versendet werden, bestimmt. Für Echtdaten gilt aber, dass diese auch zu Testzwecken ausschließlich über Produktivinstanzen (siehe unten) geschickt werden dürfen.

Für Daten, die an den Testdatenpool geschickt werden, gibt es keine Erhaltungsregel. Ein Testdatenpool kann ohne vorherige Absprache geleert werden. Die Daten werden spätestens nach einem Jahr gelöscht.

Der Testdatenpool wird nicht inhaltlich ausgewertet. Er soll Testungen von Datenbankoperationen des automatisierten Datenservice und die Simulation echter Rückprotokollierungen ermöglichen.

### **Echtdatenpool**

Der Echtdatenpool ist für Echtdaten bestimmt, die dem Bundesdatenpool zugeführt werden. Diese Daten werden entsprechend der Regeln für den Bundesdatenpool gepflegt. Alle Dokumentationen für den Regelbetrieb werden dem Echtdatenpool zugeführt.

### **Datenservices**

Als ein Datenservice wird eine automatisierte Datenverarbeitung unter einem von außen erreichbaren Endpunkt verstanden.

Für jede Instanz eines Datenservice stehen folgende Eigenschaften fest:

- Wer der Anbieter des Datenservice ist?
- Welchen Endpunkt er bedient?
- Welche Übertragungsarten er bedient (E-Mail)?
- Welcher öffentliche Schlüssel zu verwenden ist?
- Welche Bundesländer/Regionen bedient werden?
- Welche Einsender-Gruppen bedient werden (LE-stat)?
- Welche Spezifikation bedient wird (Spezifikationskennung)?
- Welche Module bedient werden?
- Welche Datenziele bedient werden (Testdatenpool, Echtdatenpool)?
- An welche nachfolgenden Datenservices die verarbeiteten Daten weitergereicht werden und

- Ob Datensicherheit garantiert wird?

Es wird bei den Datenservices grundsätzlich zwischen Testinstanzen und Produktivinstanzen unterschieden. Diese Unterscheidung hat wesentliche Auswirkungen auf bestimmte Eigenschaften eines Datenservice.

### **Produktivdatenservices**

Produktivdatenservices sind Datenservices, mit denen alle Datenpools erreicht werden können und deren Betreiber für die Datensicherheit garantieren muss.

Wegen der Datensicherheit dürfen Daten aus einem Produktivdatenservice auch nur an einen anderen registrierten Produktivdatenservice oder einen registrierten Leistungserbringer weitergeleitet werden.

Eine Testung kann auch über Produktivdatenservices erfolgen. Dabei ist unerheblich ob Echt- oder Testdaten verwendet werden. Ausschlaggebend ist, dass in dem Datenpaket, welches zur Testung verwendet werden soll, als Datenziel „Testdatenpool“ eingetragen ist.

### **Testdatenservices**

Ein Testdatenservice ist ein Datenservice, der von der betreibenden Einrichtung vorgehalten wird, um Daten zu Testzwecken anzunehmen und gleichzeitig um die Verarbeitung der Daten durch den Datenservice zu testen.

Ein Testdatenservice beruht auf Spezifikationen des Regelbetriebs. In Ausnahmefällen kann auch eine Spezifikation für den Testbetrieb die Grundlage bilden, wenn es keine entsprechenden Spezifikationen für die zu testenden Aspekte gibt.

Die angenommenen Daten dürfen unabhängig von der Ausweisung des Datenziels durch den Absender nie in einen Echtdatenpool, sondern immer nur einem Testdatenpool zugeführt werden.

Anders als die garantierte Datensicherheit des Testdatenpools (siehe oben), wird die Datensicherheit des Übertragungsweges über Testdatenservices nicht garantiert. Deswegen dürfen an Testdatenservices immer nur Testdaten (also generierte Daten ohne Bezug zu realen Personen) geliefert werden.

Im Unterschied zum Produktivdatenservice können Prüfungen im Testdatenservice deaktiviert sein.

Es wird empfohlen die Testinstanzen von der DAS ab zwei Monaten nach Veröffentlichung der ersten Spezifikation eines Spezifikationsjahres (Release V01) bereitzustellen. Im Rahmen aller weiteren Releases für ein Spezifikationsjahr wird die Bereitstellung der jeweiligen Testinstanz nach einer Woche empfohlen.

## Datenservices und Testdatenservices im Datenfluss

Datenpakete dürfen von Testinstanzen nur an nachgeschaltete Testinstanzen anderer Einrichtungen weitergereicht werden. Damit wird verhindert, dass versehentlich Testdaten in Echt Datenpools aufgenommen werden.

Datenpakete dürfen von Produktivinstanzen nur an nachgeschaltete Produktivinstanzen anderer Einrichtungen weitergeleitet werden. Damit wird verhindert, dass mögliche schutzbedürftige QS-Daten in Testinstanzen ohne Datenschutzgarantie verarbeitet werden.

## Testbetrieb mit Testsystemen und Testungen mit Produktivinstanzen

Testbetriebe finden auf der Grundlage einer Spezifikation für den Regelbetrieb mit Testinstanzen statt.

In Ausnahmen kann als Grundlage für einen Testbetrieb auch eine eigene Spezifikation erstellt werden (Betriebsart TB), wenn z. B. Änderungen in Exportverfahren unverbindlich getestet werden sollen.

Testungen mit Produktivinstanzen können jederzeit mit Test- und mit Echt Daten durchgeführt werden. Dazu muss zwingend das Datenziel „Testdatenpool“ angegeben werden.

## Regelbetrieb

Der Regelbetrieb erfordert eine eigene Spezifikation.

## 5.3 Datenversand

Folgend werden die einzelnen Datenflüsse dargestellt. Die Exportfristen pro Exportmodul sind der Tabelle `Exportmodul` der QSDOK Datenbank zu entnehmen.

### 5.3.1 Datenversand via E-Mail von den Krankenhäusern an das IQTIG

Für die Übermittlung der Daten per E-Mail muss die Exportdatei (siehe auch Abschnitt 5.2.2) in eine verschlüsselte Transaktionsdatei umgewandelt werden. Für die Verschlüsselung kann das vom IQTIG bereitgestellte Tool TPACKER (Kommandozeile) oder GPACKER (Grafische Oberfläche) genutzt werden. Im Rahmen der Strukturfrage muss der Modus K-Mode für die Verschlüsselung der Exportdatei verwendet werden. Der P-Mode ist hingegen für die Entschlüsselung des Datenflussprotokolls zu verwenden.

Das Verschlüsselungspaket kann auf der IQTIG Website heruntergeladen werden (<https://iqtig.org/spezifikationen/ergaenzende-downloads/verschluesselung/> -> Öffentliche Schlüssel zur XML-Verschlüsselung (GPACKER, XPACKER) -> xml\_schluessel -> Bundesauswertungsstelle -> Pub\_key\_Bundesauswertungsstelle\_IQTIG.pub).

Alle hierfür notwendigen Informationen werden im zugehörigen Benutzerhandbuch des Verschlüsselungspakets erläutert.

### 5.3.2 Datenversand via E-Mail von den Krankenhäusern an die Landesaufsichtsbehörden

Der Datenexport sollte automatisch durch die Software durchgeführt werden können.

Die Datenlieferfristen sind für die Module der Strukturabfragen in der jeweiligen Richtlinie und den jeweils zugehörigen Zusatzdokumenten oder Datenbanken festgelegt.

### 5.3.3 Datenversand über das E-Mail-Verfahren gemäß GGT an die LVKK/EK

Der Export der Strukturabfragedaten erfolgt auf dem gleichen Wege und nach dem gleichen Dateinamensschema wie bereits in den beiden vorangegangenen Abschnitten beschrieben.

Die XML-Exportdatei stellt im Sinne der GGT die Nutzdaten dar. Beim Datenversand gemäß GGT wird das IQTIG Verschlüsselungsprogramm (G/TPacker) nicht angewendet. Der weitere Verarbeitungsprozess, der für die Signatur, Verschlüsselung und den Versand notwendig ist, wird detailliert in Abschnitt 5.4.2 beschrieben.

## 5.4 Datenflüsse

### 5.4.1 Datenfluss der QS-Daten an das IQTIG

Die Daten werden vom Leistungserbringer direkt an die BAS als zuständige DAS gesendet.

Die folgende Abbildung 3 zeigt beispielhaft den Datenfluss an das IQTIG für die jährlichen Strukturabfragen gemäß den Richtlinien QFR, PPP und QSFFx:

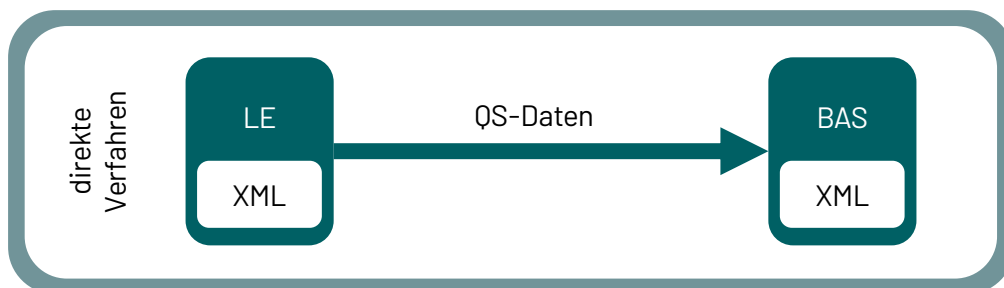


Abbildung 3: Übersicht direkter Datenfluss an die BAS

Mit den Daten eines oder mehrerer abgeschlossener Module wird vom Dokumentationssystem eine XML-Exportdatei erzeugt, in eine Transaktionsdatei komprimiert und als verschlüsselter E-Mail-Anhang an die DAS übermittelt (siehe hierzu auch Abschnitt 5.3.1). Als Zeichensatzkodierung ist UTF-8 ohne BOM mit der Eingrenzung auf "LATEINISCHE ZEICHEN IN UNICODE" zu verwenden.

Der Leistungserbringer muss beim Export darauf achten, dass nur diejenigen Module in einem Dokument zusammengefasst werden, die demselben Datenfluss zugeordnet sind.

#### 5.4.1.1 Datenfluss der Rückprotokolle

Mit dem Abschluss der Datenverarbeitung in der BAS erfolgt eine Rückprotokollierung – das sog. Datenflussprotokoll – durch die BAS direkt zum Leistungserbringer (siehe Abbildung 4).

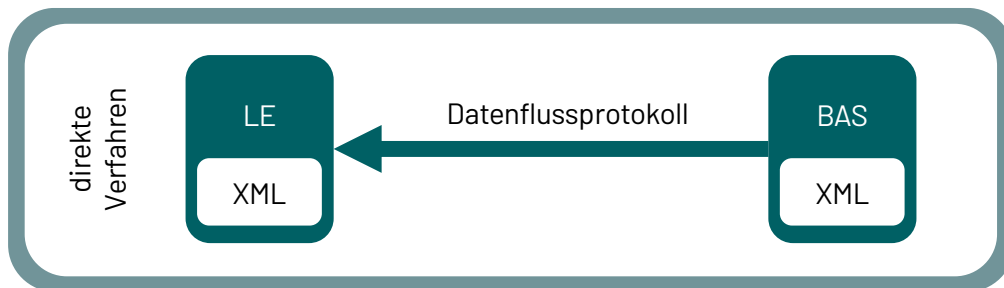


Abbildung 4: Datenfluss der Rückprotokollierung

#### 5.4.1.2 Konformitätserklärung

Die Richtigkeit der Angaben in den Datenlieferungen an das IQTIG ist durch die Leistungserbringer mittels der Übermittlung einer ausgefüllten und unterzeichneten Konformitätserklärung zu bestätigen. Beim Verfahren nach PPP-RL ist die Richtigkeit der Angaben zusätzlich auch an die LVKK/EK und die LAB zu bestätigen. Details zum Prozess finden sich in den richtlinienspezifischen Zusatzdokumenten (verfahrensspezifische technische Dokumentationen).

#### 5.4.2 Datenfluss der QS-Daten an die Landesverbände der Krankenkassen und Ersatzkassen

Die Übermittlung der Daten erfolgt gemäß der Gemeinsamen Grundsätze Technik für die elektronische Datenübermittlung gemäß § 95 SGB IV und den entsprechenden Technischen Anlagen<sup>7</sup>. Es wird das Krankenkassenkommunikationssystem (KKS) mittels E-Mail-Verfahren genutzt.

Für die Verschlüsselung wird das Verschlüsselungsverfahren PKCS#7, wie in der aktuellen Fassung der „Security Schnittstelle für das Gesundheitswesen“ beschrieben, angewendet.



##### Achtung

Das Krankenhaus muss den Exportdatensatz an jeden jeweils zuständigen Landesverband der Krankenkassen bezogen auf den Krankenhausstandort separat übermitteln, auch wenn die Datenannahmestelle für mehrere Empfänger identisch ist. Die Datenannahmestelle übernimmt nicht die Verteilung der Datenlieferungen an unterschiedliche Adressaten.

#### 5.4.2.1 Umgang mit Umlaut-Domains und E-Mail-Adressen

Die Nutzung von Umlauten ist durch die Internationalized Domain Names (IDN) möglich. Domainnamen wie z. B. <http://www.qualitätsklinikverbund.de/> sind damit nutzbar. Da die Verwendung von Zeichen aus einem Unicode-Zeichensatz in Domainnamen und E-Mail-Adressen immer noch

<sup>7</sup> <https://gkv-ag.de/datenaustausch/technische-grundsaeetze/>

Probleme in Anwendungsprogrammen bereiten, wurde das ASCII Compatible Encoding (ACE) entwickelt, in dem Regeln zur Überführung von Unicode-Zeichen in ASCII-Zeichen festgelegt wurden. Zu jeder Umlautdomain (IDN-Domain) wird parallel immer eine ACE-Domain registriert. Nähere Informationen findet man z.B. bei Wikipedia<sup>8</sup>. Für das oben genannte Beispiel <http://www.qualitätsklinikverbund.de/> würde der ACE-Name xn--qualitätsklinikverbund-b2b.de lauten.

E-Mail-Adressen dürfen im lokalen Teil (vor dem @-Zeichen) keine Umlaute enthalten. Die E-Mail-Adresse müller@muster.de wäre nicht möglich. Möglich ist lediglich mueller@muster.de. Es existieren experimentelle Request for Comments (RFCs)<sup>9</sup>, in denen UTF-8-codierte, internationale E-Mail-Adressen ermöglicht werden. Dies wird jedoch nicht flächendeckend im Produktiveinsatz genutzt, aus diesem Grund wird in dieser Spezifikation von der Annahme ausgegangen, dass der lokale Teil der E-Mail-Adresse immer durch ASCII-Zeichen repräsentiert wird.

In den E-Mail-Elementen des XML-Schemas sowie in der Auftragsdatei sind ausschließlich Zeichen des ASCII-Zeichensatzes erlaubt. E-Mail-Adressen mit einem IDN Domainnamen sind entsprechend den ACE-Regeln durch ASCII-Zeichen darzustellen und zu übermitteln.

#### 5.4.2.2 Übermittlungsdateien

Die Übermittlung von Daten im KKS erfolgt immer durch zwei Dateien, die Nutzdatendatei und die Auftragsdatei. Im Fall der QFR-, QSFFx- und PPP-Richtlinie ist die Nutzdatendatei die XML Exportdatei, in der das Nachweisverfahren dokumentiert wird.

##### **Nutzdatendatei**

Die Nutzdatendatei ist binärkodiert und kann grundsätzlich einen beliebigen Inhalt haben. In der vorliegenden Spezifikation ist dieser durch ein XML-Schema genormt.

Gemäß der Gemeinsamen Grundsätze Technik Abschnitt 5 (Sicherheitsverfahren) ist die Nutzung von Verschlüsselungsverfahren und digitalen Signaturen (Security Schnittstelle SECON) verpflichtend anzuwenden, um die Vertraulichkeit, Integrität und Verbindlichkeit der zu übermittelnden Daten zu gewährleisten. Demnach ist die XML-Exportdatei mit dem privaten Schlüssel des Datensenders zu signieren und anschließend mit dem öffentlichen Schlüssel des Datenempfängers zu verschlüsseln. Der Versand und die Verschlüsselung erfolgen mit dem Institutionskennzeichen, welches im Auftragsdatensatz als „Absender\_Eigner“ hinterlegt ist. An dieses IK werden auch die technischen Rückmeldungen gesendet. Gegen dieses Zertifikat wird die Antwort verschlüsselt. Die IK-Nummer in `/root/header/ggt/IK_des_Absenders/@V` im Nutzdatensatz und „Absender\_Eigner“ im Auftragsdatensatz müssen identisch sein. Der Datenempfänger muss bei Datenannahme das Zertifikat des Absenders auf Gültigkeit prüfen. Die Krankenhäuser verwenden Zertifikate der TrustCenter ITSG und DKTIG. Die Zertifikate sind auf die IK-Nummern

---

<sup>8</sup> [https://de.wikipedia.org/wiki/Internationalisierter\\_Domainname](https://de.wikipedia.org/wiki/Internationalisierter_Domainname)

<sup>9</sup> <https://tools.ietf.org/html/rfc5336>

ausgestellt und auf einer Webseite des ITSG immer in einer aktuellen Fassung verfügbar. In der Auftragsdatei werden immer sowohl die physikalische als auch die logische IKNR eingetragen. Die Verschlüsselung der Nutzdatendatei erfolgt immer mit dem Zertifikat der logischen IKNR (EMPFÄNGER\_NUTZER). Für eine standortbasierte Datenübermittlung sind keine Standortzertifikate notwendig. Die Signierung der Daten erfolgt mit dem Zertifikat des Krankenhauses für alle Standorte. Jedoch ist beim Versand darauf zu achten, dass der Versand an die Landesverbände der Krankenkassen und an die Ersatzkassen immer standortbezogen, einzeln erfolgt. Bei der Absender-IK muss es sich nicht zwingend um das dem leistungserbringenden Standort zugehörige Institutionskennzeichen des Krankenhauses handeln.

ABSENDER\_EIGNER

muss

/root/header/ggt/IK\_des\_Absenders/@V

entsprechen,

//ggt/IK\_des\_Absenders/@V

und

/root/body[1]/data\_container[1]/care\_provider[1]/IKNRKH[1]/@V

können hingegen unterschiedlich sein. Der Versand der Daten kann auch von einer anderen IK (z. B. einem externen Anbieter) erfolgen. Die technische Rückmeldung der Landesverbände der Krankenkassen und Ersatzkassen werden hierbei wieder an die Absender-IK erfolgen.

### Dateinamenskonventionen

Der Dateiname der Nutzdatendatei und der Auftragsdatei spielen bei der Übermittlung und dem Empfang eine wichtige Rolle für die Identifikation in der Datenannahmestelle. In diesem Dateinamen muss die Verfahrenskennung enthalten sein. Im Folgenden wird dies anhand der Verfahrenskennung für die Strukturabfragen nach QSFFx-RL, PPP-RL und QFR-RL exemplarisch dargestellt. Gemäß den Richtlinien des KKS wird der 8-stellige physische Dateiname für die Übertragung nach dem folgenden Schema gebildet:

<[E/T]><XXXX><999>

Erläuterung der Dateinamensbausteine:

- <[E/T]> → Echtdaten (E) oder Testdaten (T)
- <XXXX> → 4-stellige Verfahrenskennung inkl. Versionsnummer („FFX0“ bzw. „PPP0“ bzw. „QFR0“)
- <999> → 3-stellige laufende Transfernummer bei der Übertragung zwischen zwei direkt verbundenen Kommunikationspartnern (Transfernummer)

Der Name der zugehörigen Auftragssatzdatei wird aus dem Dateinamen der Transferdatei und dem Zusatz (Dateiendung) „.AUF“ gebildet.



Beispiel zweier zusammengehöriger Dateien für einen Übertragungstest:

- TFFX0001
- TFFX0001.AUF

Die Transfernummer wird vom Sender bei jedem neuen Datenexport inkrementiert. Sie wird auch bei jeder, im Datenfluss liegenden Zwischenstation (Hop), inkrementiert. D. h. der physische Dateiname ändert sich bei jedem Hop, den die Datenlieferung auf dem Weg zum Empfänger passiert.

### Auftragsdatei

Der Aufbau der Auftragsdatei ist in den „Richtlinien für den Datenaustausch im Gesundheits- und Sozialwesen“ beschrieben. Folgende Inhalte werden durch die vorliegende Spezifikation vorgegeben.

VERFAHREN\_KENNUNG (5-stellig):

- <[E/T]> → Echtdaten (E) oder Testdaten (T)
- <XXXX> → 4-stellige Verfahrenskennung inkl. Versionsnummer (hier immer „FFX0“)

Zur Übermittlung von Daten im KKS wird eine Verfahrenskennung (Gemeinsame Grundsätze Anlage 4) benötigt. Die Verfahrenskennung im Rahmen der QSFFx Richtlinie lautet „FFX“. Die Verfahrenskennung im Rahmen der PPP Richtlinie lautet „PPP“. Die Verfahrenskennung im Rahmen der QFR Richtlinie lautet „QFR“. Der Verfahrenskennung folgt eine 0 als Versionsnummer.

Logischer Dateiname (11-stellig)<sup>10</sup>: <XXX><NV><YYYYYY>

Erläuterung der Dateinamensbausteine:

- <XXX> Verfahrenskennung (FFX/PPP/QFR)
- <NV> Nachweisverfahren (NV)
- <YYYYYY> 6-stelliges Standortkennzeichen des Krankenhauses (z. B. 771234)

Beispiel: <FFXNV771234>

VERFAHREN\_KENNUNG\_SPEZIFIKATION (5-stellig)

EXPKK → Übertragung des Exportdatensatzes von KH an LVKK/EK

ANTKK → Übertragung der Rückmeldung (Antwortdatei) von LVKK/EK an KH

Empfänger-IKNR (physikalisch, logisch)

---

<sup>10</sup> Bezeichnung laut GGT, Anlage 2 „Auftragsdatei (Auftragssatz)“: DATEINAME

Datenannahmestellen haben teilweise unterschiedliche physikalische und logische IK-Nummern. Die physikalische IKNR stellt eine Zwischenstation im Datenfluss dar und wird in der Auftragsdatei im Element `EMPFÄNGER_PHYSIKALISCH` hinterlegt. Die logische IKNR repräsentiert den finalen Empfänger der Nachricht und wird im Feld `EMPFÄNGER_NUTZER` hinterlegt. Die Verschlüsselung der Nutzdaten erfolgt mit dem öffentlichen Schlüssel der logischen IKNR. `EMPFÄNGER_PHYSIKALISCH` ist eine empfangende Zwischenstation, kann aber im Unterschied zu `EMPFÄNGER_NUTZER` die Nutzdatendatei nicht entschlüsseln.

#### KOMPRIMIERUNG (2-stellig)

Die Dateien können komprimiert übermittelt werden. Als Komprimierungsverfahren ist das Produkt „gzip“ zulässig. Der fakultative Einsatz einer Komprimierung einer Transferdatei muss im zugehörigen Auftragsatz im Feld `KOMPRIMIERUNG` angegeben werden. Die zulässigen Werte lauten:

- 00 → keine Komprimierung
- 02 → gzip

Das Komprimierungsverfahren bzw. dessen Einsatz sind immer bilateral zwischen dem Absender und dem Empfänger abzustimmen.

#### VERSCHLUESSELUNGSART (2-stellig) + ELEKTRONISCHE\_UNTERSCHRIFT (2-stellig)

- 03 + 03 → PKCS#7-Format

Es sind ausschließlich signierte und verschlüsselte Nachrichten zulässig.

#### E-MAIL-ADRESSE ABSENDER (variable Länge)

In diesem Feld ist die E-Mail-Adresse der Einrichtung, des Standorts oder des Krankenhauses anzugeben, die die elektronische Kommunikation mit den jeweils zuständigen Landesverbänden der Krankenkassen durchführt. An diese E-Mail-Adresse sind die Antworten der jeweils zuständigen Landesverbände im Sinne des Datenflussprotokolls (siehe Abschnitt 6) zu übertragen. Bitte beachten Sie die Vorgaben zu E-Mail-Adressen in Abschnitt 5.4.2.1).



#### **Achtung**

Die E-Mail-Adresse in `/root/header/ggt/TechnischeEmail/@V` muss mit dem Wert für `E-MAIL-ADRESSE ABSENDER` im Auftragsdatensatz identisch sein.

---

## Rückmeldung ans Krankenhaus vom jeweils zuständigen Landesverband der Krankenkassen

Die jeweils zuständigen Landesverbände der Krankenkassen bestätigen gegenüber den Krankenhausträgern bzw. Krankenhäusern den Erhalt und die technische Lesbarkeit der übertragenen Nutzdaten in Form eines Datenflussprotokolls (siehe Abschnitt 6). Diese werden nach Verarbeitung auf gleichem Übertragungsweg wie die Übertragung der Daten übermittelt. Die Beschreibung der XML-Struktur ist in Abschnitt 10 erläutert.

Liegt bei einem Datenempfänger ein Fehlerfall vor, so ist dieser **gemeinsam** mit dem Sender der Daten zu überprüfen:

- Fehler beim Empfänger (LVKK): Korrektur des Fehlers in der Datenannahme und erneutes Einlesen der bereits übermittelten Exportdatei.
- Fehler beim Sender (KH): Korrektur des Fehlers in der Software und erneuter Versand eines korrekten Datenexports an **ALLE** Empfänger. Sollte in diesem Fall der ursprüngliche Export von einem oder mehreren Datenannahmestellen angenommen und fehlerfrei protokolliert worden sein, ist hier ebenfalls eine Korrektur (des nicht erkannten Fehlers) in der Datenannahme vorzunehmen! Die betroffenen Datenannahmestellen sind durch das Krankenhaus entsprechend zu informieren. Beim erneuten Export ist der Datensatz mit einer höheren Versionsnummer in den XML Nutzdaten zu versenden.

Die zuständigen Kontaktdaten der Landesverbände der Krankenkassen finden sich in den Datenserviceinformationen (siehe Abschnitt 9.7.2) sowie der Übersicht über die Datenannahmestellen bei den Landesverbänden der Krankenkassen und der Ersatzkassen.



### Hinweis

Zum Zeitpunkt der Veröffentlichung dieser Technischen Dokumentation sind die zuständigen Datenannahmestellen noch nicht bekannt. Diese werden sobald bekannt in den Datenserviceinformationen (siehe Abschnitt 9.7.2) veröffentlicht.

---

#### 5.4.2.3 E-Mail-Verfahren

Die Nutzdatendatei sowie die Auftragsdatei werden nach Erstellung mittels E-Mail-Verfahren vom Krankenhausträger bzw. dem Krankenhaus an die jeweils zuständigen Landesverbände der Krankenkassen übermittelt. Die entsprechenden Festlegungen sind in den Gemeinsamen Grundsätzen Technik geregelt.

#### 5.4.2.4 Datenannahmestellen bei den Landesverbänden der Krankenkassen und Ersatzkassen

Die zuständigen Datenannahmestellen der Landesverbände der Krankenkassen und Ersatzkassen werden in den Datenserviceinformationen aufgelistet.

Die hier aufgeführten Adressaten entsprechen den technischen Ansprechpartnern, mit denen die elektronische Kommunikation (Datenübermittlung) abgewickelt wird. Sofern keine weiteren Angaben vorliegen, sind die hinterlegten Datenannahmestellen länderübergreifend tätig. Es sind für die Kommunikation mittels GGT die technischen E-Mail-Adressen sowie die physikalischen- und logischen IK-Nummern zu nutzen (siehe Abschnitte 5.4.2.2 und 9.7.2).

#### **5.4.2.5 Bereitstellung von Testinstanzen**

Es wird empfohlen, Testinstanzen der Datenannahmestellen bei den Landesverbänden der Krankenkassen und Ersatzkassen im ersten Spezifikationsjahr neun Monate und ab dem zweiten Spezifikationsjahr drei Monate nach Veröffentlichung der ersten Spezifikation eines Spezifikationsjahres (Release V01) bereitzustellen. Im Rahmen aller Updates für ein Spezifikationsjahr wird die Bereitstellung der jeweiligen Testinstanz eine Woche nach Veröffentlichung empfohlen.

## 6 Rückprotokollierung

Nach erfolgreicher Eingangsverarbeitung durch den Empfänger erhält der Leistungserbringer auf dem Eingangskanal<sup>11</sup> ein Datenflussprotokoll. Das Datenflussprotokoll bestätigt einerseits den Erhalt der Exportdatei und enthält andererseits alle Prüfergebnisse aus der Datenvalidierung.

Das Protokoll ist von der Dokumentationssoftware einzulesen und dem Anwender darzustellen. Gegebenenfalls müssen die fehlerhaften Datensätze korrigiert und erneut übermittelt werden.

Sollte der Leistungserbringer kein Datenflussprotokoll von einer belieferten Datenannahmestelle (DAS) erhalten, so muss der Leistungserbringer diese DAS kontaktieren, um deren Validierungsergebnisse bzw. Datenflussprotokoll anzufordern. Jede DAS validiert entsprechend der veröffentlichten Spezifikation. Dennoch ersetzt das Validierungsergebnis einer DAS nicht ein ausstehendes DFP einer anderen DAS.

### **Regelungen für ein Vorgehen bei Verarbeitungsabbrüchen im besonderen Fehlerfall**

Diese Regelungen dienen dazu, ein abgestimmtes Vorgehen durch alle am Datenfluss beteiligten Stellen zu etablieren, um die Datenbestände in allen Instanzen zu konsolidieren, wenn es zu unerwarteten Störungen in der Verarbeitung von Datenlieferungen bei den jeweiligen Empfängern kommt. Solche Störungen sind selten und stellen einen Ausnahmefall dar.

Solche Störungen können in äußerst seltenen Fällen aufgrund einer fehlerhaften automatisierten Verarbeitung von Datenlieferungen beim Empfänger dazu führen, dass fehlerhafte Datenflussprotokolle versandt werden. Dadurch können beim Leistungserbringer unter Umständen mehrere Datenflussprotokolle zu einer GUID eintreffen, die widersprüchliche Einstufungen der Vorgänge enthalten. Um diesen Umstand an allen am Datenfluss beteiligten Stellen aufzulösen, müssen die involvierten Vorgänge erneut verarbeitet werden.

Das präferierte Vorgehen in dieser Situation ist ein erneuter Versand aller betroffenen Datensätze, die unter der betroffenen GUID gesandt wurden, durch den Leistungserbringer in einer höheren Version. Dieser Versand muss nach Rücksprache mit der Stelle, bei der die fehlerhafte Verarbeitung aufgetreten ist, geschehen und wird deshalb telefonisch von den Empfängern beim betroffenen Leistungserbringer initiiert. Für den erneuten Versand ist beim Leistungserbringer ein Export aller betroffenen Vorgangsnummern einschließlich der Erhöhung der zugehörigen Versionsnummer unter einer neuen GUID durchzuführen. Das im Zuge des Exports erstellte XML-Dokument erhält alle Vorgänge als Update in der jeweils höchsten Versionsnummer, die das Softwaresystem beim Leistungserbringer erstellt, und erhält eine neue und damit unverbrauchte GUID. Somit können alle Datensätze im Datenfluss einschließlich der zugehörigen Rückprotokollierung mittels des Datenflussprotokolls bis hin zum Leistungserbringer regulär verarbeitet werden.

---

<sup>11</sup> Es können Abweichungen auftreten. Nähere Informationen sind bei der zuständigen DAS einzuholen.

Sollte der Leistungserbringer nicht in der Lage sein, die Daten erneut zu exportieren, so ist der alternative Weg zu wählen, bei dem ein bereits verarbeitetes Dokument (einschließlich Rückprotokollierung mittels Datenflussprotokoll bis hin zum Leistungserbringer) – und somit eine bereits verarbeitete GUID – erneut verarbeitet wird. Das genaue Vorgehen ist im Abschnitt 6.1 beschrieben.

## 6.1 Funktion, Aufbau und Erstellung

In diesem Kapitel wird die Rückprotokollierung in Bezug auf die Funktion, den Aufbau und die Erstellung beschrieben.

### 6.1.1 Funktion des Datenflussprotokolls im Datenfluss

Ein Datenflussprotokoll wird erstellt, wenn das Dokument keine weitere Verarbeitung mehr erlaubt. Das ist dann der Fall, wenn das Dokument durch einen der vorgesehenen Prüfungs- und Verarbeitungsschritte den Status `ERROR` erhält oder wenn das Dokument in der Bundesauswertungsstelle, den Landesverbänden der Krankenkassen und Ersatzkassen oder den Landesaufsichtsbehörden vollständig und erfolgreich verarbeitet wurde und den Status `WARNING` oder `OK` trägt.

Das Datenflussprotokoll dokumentiert alle an dem Dokument durchgeführten Prüfungen und deren Ergebnisse.

Ein Datenflussprotokoll wird in der Regel bis zum Leistungserbringer zurückübermittelt.

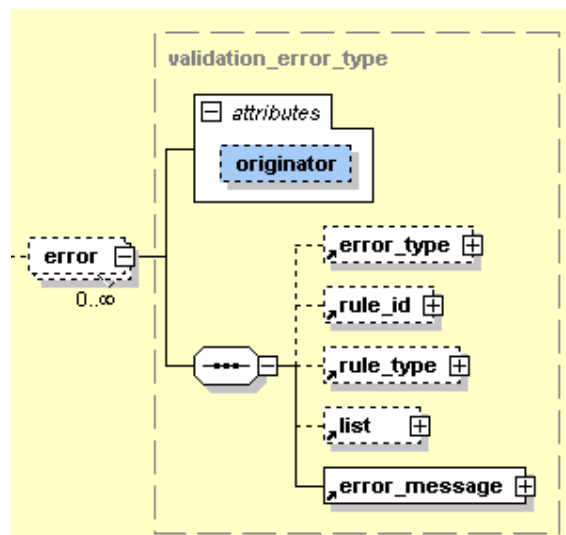


Abbildung 5: Attribut "originator" im Prüfungs- und Fehlerprotokoll

### 6.1.2 Aufbau und Erstellung

Die vorgenommenen Prüfungen werden in den dafür vorgesehenen Bereichen im XML-Code des übermittelten Dokuments protokolliert. Das Protokoll des Dokuments wächst damit mit jeder Prüfung an.

Nachdem alle Prüfungen der datenentgegennehmenden Stelle abgeschlossen sind, wird für die Rückprotokollierung der Prüfungsergebnisse eine Kopie des Dokuments von allen QS Daten (Element `<qs_data>`) befreit. Das verbleibende XML enthält innerhalb der ursprünglichen Struktur des Dokuments die bis dahin protokollierten Prüfungen und die sich daraus ergebenden Statusmeldungen der Datensätze und des Dokuments.

Bezogen auf einen bestimmten Datensatz ist es erst nach der letzten abgeschlossenen Prüfung auf Ebene der DAS möglich, eine Aussage darüber zu treffen, ob sich dieser Datensatz für die Aufnahme in den Datenpool eignet oder nicht.

Um dem Leistungserbringer das konkrete Ergebnis seiner Datenlieferung in Bezug auf den Datenpool mitteilen zu können, wird auch das bis zum Schluss weitergeführte Dokument von QS Daten befreit und als Datenflussprotokoll an den Leistungserbringer übermittelt.

### **Miniprotokoll**

Die Erstellung eines vollständigen Datenflussprotokolls ist nur möglich, wenn die vom Leistungserbringer gelieferte Exportdatei schemakonform ist. Andernfalls ist nur ein reduziertes Protokoll („Miniprotokoll“) möglich. Im Unterschied zum vollständigen Datenflussprotokoll enthält das Miniprotokoll nicht alle Validierungsergebnisse, denn Schemaverletzungen würden keine Validierung auf Datensebene erlauben.

### **Die Fehlermeldungen**

Für jeden Fehler wird wenigstens das Element `<error_message>` ausgefüllt. Andere Elemente bleiben bei einzelnen Fehlerarten leer. Tabelle 5 gibt einen Überblick darüber, unter welchen Bedingungen in den Feldern der Fehlerdatei Angaben erforderlich sind.

Die Bogenliste `<list>` umfasst einen oder mehrere Namen von Teildatensätzen, welche einen Bezug zu einer Regel haben. Entscheidend für den Bogenbezug sind die in der Tabelle `Regeln` formulierten Regeln, nicht die für den Exportdatensatz umformulierten Regeln.

Die Bogenfeldliste umfasst einen oder mehrere Namen von Bogenfeldern, welche einen Bezug zum Fehler haben. Bei der Fehlerart WERT enthält die Liste nur ein Element. Der Bogenfeldname umfasst auch den Namen des zugehörigen Teildatensatzes. Der Bezug zum Modul kann entfallen, da dieses über die Vorgangsnummer identifiziert werden kann.

Für jede Regel gibt es eine Liste von Bogenfeldern, identifiziert über die Feldnamen der Regeln. Damit die Liste nicht durch Parsen ermittelt werden muss, wird sie auch über die Tabelle `Regel-Felder` zur Verfügung gestellt. Über die Regelnummer können die Teildatensätze, welche Bezug zu einer Regel haben, durch folgende Abfrage identifiziert werden:

```
SELECT DISTINCT Bogen.name FROM (Modul INNER JOIN (Feld INNER JOIN
(Bogen INNER JOIN BogenFeld ON Bogen.idBogen = BogenFeld.fkBogen) ON
Feld.idFeld = BogenFeld.fkFeld) ON Modul.idModul = Bogen.fkModul)
INNER JOIN RegelFelder ON BogenFeld.idBogenFeld = RegelFelder.fkBogenFeld
WHERE RegelFelder.fkRegeln=<Regelnummer>;
```

Für die Regelnummer `<rule_id>` ist die entsprechende Nummer (Attribut `idRegeln`) der Tabelle `Regeln` anzugeben.

Bei Teildatensätzen, welche mehrfach angelegt werden können, muss die Nummer des betreffenden Teildatensatzes in eckigen Klammern angehängt werden (z. B. `P[1]`, `P[2]` usw.). „Nummer des betreffenden Teildatensatzes“ ist die Position des Teildatensatzes im XML.

Tabelle 5: Ausfüllen der Elemente eines Validation-Items in Abhängigkeit von den Fehlerarten

Feld	Fehlerart	Regelnr	Regeltyp	Liste	Meldung
Element (xml)	<code>&lt;error_type&gt;</code>	<code>&lt;rule_id&gt;</code>	<code>&lt;rule_type&gt;</code>	<code>&lt;list&gt;</code>	<code>&lt;error_message&gt;</code>
	EXPORT	Ja	-	<code>&lt;Bogen&gt;</code>	ja
	DOPPELT	Ja	-	-	ja
	TDS	Ja	-	<code>&lt;Bogenliste&gt;</code>	ja
	WERT	Ja	-	<code>&lt;Bogenfeldliste&gt;</code> <sup>12</sup>	ja
	REGEL	Ja	ja	<code>&lt;Bogenliste&gt;</code>	ja

Um die Fehleranalyse zu vereinfachen, wurden die potenziellen Fehler in Kategorien (XML-Element `<error_type>`) unterteilt, die bestimmte Prüfprozesse (XML-Element `<validation_item>`) durchlaufen. Die konkreten Fehlermeldungen sind in der Spezifikationsdatenbank QSDOK hinterlegt:

- Tabelle `Regeln.meldung`: enthält spezifische Fehlermeldungen bei entsprechenden Regelverletzungen
- Tabelle `Fehlermeldung.meldung`: enthält standardisierte und allgemeingültige Fehlermeldungen. Folgende Tabelle zeigt Beispiele bei bestimmten Fehlerfällen.

<sup>12</sup> In der Regel wird hier nur ein Bogenfeld aufgeführt. Ausnahme ist, wenn Kombinationsfelder geprüft werden.



Tabelle 6: Beispiele von Fehlermeldungen

Fehlerart	Standardisierte Meldung	Beschreibung
Standardisierung der Meldungen bei Bestätigungstatus mit Fehlerart TDS.	Erforderlicher Teildatensatz <Bogen.name> („<Bogen.bezeichnung>“) existiert nicht.	Wenn ein obligatorischer Teildatensatz (Attribut <code>Bogen.fkBogenZahl</code> + oder 1 ist, oder ein zu einem Kind-Teildatensatz zugehöriger Mutterteildatensatz) eines Vorgangs in den Exportdateien einer Transaktion nicht vorkommt
	Die Angaben im Datensatz erfordern einen Teildatensatz <Bogen.name> („<Bogen.bezeichnung>“). Dieser fehlt.	Wenn die Existenzbedingung eines Kind-Teildatensatzes im zugehörigen Mutterteildatensatz erfüllt ist, aber kein Kind-Teildatensatz vorhanden ist (Abschnitt B 9.2.2).
	Die Angaben im Datensatz lassen keinen weiteren Teildatensatz <Bogen.name> („<Bogen.bezeichnung>“) zu, obwohl ein solcher übermittelt wurde.	Wenn die Existenzbedingung eines Kind-Teildatensatzes im zugehörigen Mutterteildatensatz nicht erfüllt ist, aber trotzdem ein Kind-Teildatensatz existiert (Abschnitt 9.2.2).

Die Fehlermeldungen, die nicht von der Tabelle Regel in der Access-Datenbank abgedeckt sind, sind in der Tabelle Fehlermeldung hinterlegt. Diese sollen eine Standardisierung von Fehlermeldungen und klaren Bedeutungen unterstützen.

### Prüfungsprozess und Ergebnisprotokollierung

Ausgangspunkt ist eine prinzipiell offene Anzahl von Prüfungen. Welche Prüfungen konkret durchgeführt werden, ist abhängig vom Datenfluss. Für die Protokollierung der Prüfungen und deren Ergebnisse gibt es auf Dokumentenebene im Header und auf Fallebene im `<case_admin>` das Element `<protocol>`.

Auf Dokumentenebene sind alle Prüfungen zu dokumentieren, einschließlich der Prüfungen, die ausschließlich die Datensätze betreffen. Eine prüfende Einrichtung trägt sich als `<validation_provider>` in die entsprechende Auflistung ein und dokumentiert dann ihre durchgeführten Prüfungen in der Auflistung `<validation_item>` (Ausnahme: Prüft der Leistungserbringer, sind in keinem Fall die Daten des Leistungserbringers einzutragen. In diesem Fall wird der Softwareanbieter als `<validation_provider>` eingetragen).

Prüfungen, die – wie z. B. die Schemakonformität – das Dokument insgesamt betreffen, sind ausschließlich im Headerbereich einzutragen.

Prüfungen, die – wie zum Beispiel die Prüfung auf Plausibilitätsregeln – auf Fallebene erfolgen, müssen folgendermaßen protokolliert werden:

- Das Ergebnis in Bezug auf das gesamte Dokument muss im `<header>` eingetragen werden.
- Das Ergebnis der Fallprüfung muss in `<case_admin>` eingetragen werden, sofern der Status dieser Prüfung nicht OK ist (siehe auch unten).
- Alle Ergebnisse einer Prüfung, die auf Fallebene erfolgt, müssen mit einer gemeinsamen, dokumentweit eindeutigen ID im Attribut ID des Elements `<validation_item>` eingetragen werden. Dadurch ist es möglich, über die ID eines Prüfungsergebnisses, die man auf Fallebene findet, auf Dokumentenebene den `<validation_provider>` eindeutig zu identifizieren.

Zur Veranschaulichung dieser Konstruktion soll im Folgenden eine Analogie aus dem relationalen DB-Modell bemüht werden. So kann die Dokumentenebene als Master-Tabelle und die Fallebene als Detail-Tabelle bezeichnet werden. Letztere enthält die zum Master gehörenden Detail-Datensätze, auf die über das Attribut „ID“ referenziert werden kann.

### Prüfungsergebnisse

Prinzipiell wird als Ergebnis jeder Prüfung eine der folgenden Aussagen über das geprüfte Objekt getroffen:

- Keine Auffälligkeiten
- Auffälligkeiten, die einer Weiterverarbeitung nicht im Weg stehen
- Auffälligkeiten bzw. Fehler, die eine Weiterverarbeitung des Datensatzes ausschließen. Eine Korrektur und erneute Lieferung des Datensatzes sind erforderlich.

In der datentechnischen Übersetzung wird dieses durch:

- OK
- WARNING
- ERROR

ausgedrückt, die das Ergebnis der Prüfung im Attribut „V“ des Elements `<status>` im Element `<validation_item>` zusammenfassen.

Darüber hinaus gibt es die Möglichkeit, eine beliebige Anzahl von `<error>` Elementen mit einer `<error_message>` im `<status>` Element unterzubringen.

Im Fall einer Auffälligkeit muss wenigstens eine standardisierte Fehlermeldung im `<status>` Element der von der Prüfung betroffenen Ebene (Vorgang oder Dokument) untergebracht werden (Abschnitt 9.7.3).

### Beziehungen Vorgangsebene/Dokumentenebene

Es gibt zwei Kategorien geprüfter Objekte, die zueinander in einer hierarchischen Beziehung stehen:

- Erste Hierarchieebene: das gesamte Dokument
- Zweite Hierarchieebene: der Fall

Jedes dieser Objekte hat einen Status in Bezug auf die Weiterverarbeitung, der sich auf die Ergebnisse der durchgeführten Prüfungen bezieht.

Auf Dokumentenebene ist dieser Status im Unterelement `<status_document>` von `<protocol>` im Attribut `@v` abgelegt.

Auf Fallebene ist dieser Status ebenfalls in einem Attribut `@v` eines Unterelements von `<protocol>` abgelegt, welches hier aber `<status_case>` benannt wird.

In Bezug auf die Weiterverarbeitung gibt es folgende Regeln:

Ein `ERROR` in einer der Prüfungen verhindert die Weiterverarbeitung des geprüften Objekts. Eine oder mehrere Auffälligkeiten (`WARNING`, `ERROR`) auf Fallebene bedeuten ein `WARNING` in dem korrespondierenden Eintrag auf Dokumentenebene. Wenn bei einer fallbezogenen Validierung in allen Fällen auf Status `ERROR` erkannt wird, muss auch für das Dokument abweichend von der Regel unter 2. der korrespondierende Eintrag auf Dokumentenebene auf `ERROR` gesetzt werden. Der Status (`<status_case>/<status_document>`) eines Objekts kann nicht „besser“ sein als sein schlechtestes Prüfergebnis.

## Szenarien

Aus diesen Regeln abgeleitet, soll der Status jedes geprüften Objekts nach jeder Prüfung entsprechend dem Prüfergebnis aktualisiert werden. Daraus ergibt sich folgender Aktualisierungs- und Protokollierungsplan:

### Vor der Prüfung und Protokollierung

- (1) → Feststellen der höchsten ID in Bezug auf vorhandene `<validation_item>`-Elemente.
- (2) → Festgestellte ID um 1 inkrementieren und als ID der anstehenden Prüfung festlegen.

### Protokollierung der fallbezogenen Prüfung

Nachdem die fallbezogene Prüfung erfolgt ist, ist dies auf der Fallebene und der Dokumentenebene folgendermaßen zu protokollieren:

#### Protokollierung auf Fallebene

Positive Prüfungen werden auf Fallebene nicht protokolliert. Wenn eine Prüfung auf Fallebene keine Auffälligkeit feststellt, wird dieses Ergebnis nicht dokumentiert. Das Ergebnis OK ist implizit anzunehmen, wenn kein Fehler protokolliert wurde.

Auf Fallebene wird nur dann protokolliert, wenn bei der Prüfung eine Auffälligkeit festgestellt wurde. Falls eine Auffälligkeit festgestellt wird, sind die Schritte 3 bis 6 abzuarbeiten.

- (3) → `<validation_item>` der Liste hinzufügen, dabei die unter 2. ermittelte ID verwenden.
- (4) → `<status_case>` des Falls auslesen.

(5) → Ergebnis der Prüfung mit dem Status des Falls vergleichen. In den Fällen, bei denen das Ergebnis der Prüfung schlechter ist als der aktuelle Status des Falls, wird der Status mit dem Ergebnis der Prüfung aktualisiert.

(6) → Falls ein Ergebnis der Prüfung schlechter ist als „OK“, muss dieses als dokumentbezogenes Ergebnis „WARNING“ vermerkt werden.

#### Protokollierung auf Dokumentenebene

(7) → <validation\_item> mit dem unter 4. ermittelten Prüfungsergebnis der fallbasierten Prüfung unterhalb des Elements <validation\_provider> eintragen. Falls <validation\_provider> für die eigene Einrichtung noch nicht besteht, muss er angelegt werden.

(8) → <status\_document> auslesen.

(9) → Das unter 6. ermittelte Gesamtergebnis der Prüfung muss mit dem Status des Dokuments verglichen werden. Falls das Ergebnis der Prüfung schlechter ist als der aktuelle Status des Dokuments, muss dessen Status mit dem Ergebnis der Prüfung aktualisiert werden.

#### Protokollierung der dokumentenbezogenen Prüfung

Nachdem die dokumentenbezogene Prüfung erfolgt ist, ist dies auf der Dokumentenebene folgendermaßen zu protokollieren:

(10) → <validation\_item> mit dem Prüfungsergebnis unterhalb des Elements <validation\_provider> eintragen. Falls <validation\_provider> für die eigene Einrichtung noch nicht besteht, muss er angelegt werden.

(11) → <status\_document> auslesen.

(12) → Das Ergebnis der Prüfung mit dem Status des Dokuments vergleichen und in dem Fall, in dem das Ergebnis der Prüfung schlechter ist als der aktuelle Status des Dokuments, dessen Status mit dem Ergebnis der Prüfung aktualisieren.

## 7 Prozesse in den Datenannahmestellen

In diesem Abschnitt werden die Verarbeitungsschritte in einer Datenannahmestelle (DAS) dargestellt.

Begriffe, die nachfolgend verwendet werden:

- Exportdatei:
  - gelieferte Datei des Leistungserbringers
- Datensatz:
  - Datensatz entspricht einem case-Element aus einer Exportdatei. Ein Vorgang kann adressiert werden über die Vorgangsnummer (GUID). Ein Datensatz kann mit der Vorgangsnummer (GUID) - `case/case_admin/guid/@V` - und `case/case_admin/version/@V` adressiert werden.
- Datenflussprotokoll:
  - Validierungsergebnisse für die gelieferte Exportdatei
  - ohne `./qs_data` aus der angelieferten Exportdatei
  - valide gegen Schema `datenflussprotokoll.xsd`
  - nach DPP<sup>13</sup>-Durchlauf im Ordner `.\output\protocol\`
  - Adressat: Leistungserbringer
- geprüfte Datei:
  - Datenflussprotokoll, aber mit `./qs_data` aus der angelieferten Exportdatei
  - nach DPP-Durchlauf im Ordner `.\output\files\`
  - wird nicht dem Leistungserbringer gesendet
- Miniprotokoll:
  - Protokoll von Fehlern, die ein vollständiges Datenflussprotokoll verhindern (bspw. Schema-verletzungen)
  - valide gegen Schema `datenflussprotokoll.xsd`
  - nach DPP-Durchlauf im Ordner `.\output\protocol\`
  - Adressat: Leistungserbringer

Die Datenannahmestellen müssen ganzjährig Datenlieferungen im Rahmen der QSFFx Richtlinie annehmen. Der Datenfluss vom Leistungserbringer (LE) an eine LVKK/EK erfolgt über das Krankenkassenkommunikationssystem.

---

<sup>13</sup> Datenprüfprogramm  
© IQTIG 2026

Der Zeitpunkt des Dateneingangs einer Exportdatei muss persistiert werden. Ist nach Prüfung die Exportdatei wohlgeformt und schemakonform, dann wird der Zeitstempel der Exportdatei hinzugefügt (`/root/header/document/das_receive_dttm/@V`).

Position von `root/header/document/das_receive_dttm/@V`:

```
preceding-sibling::*[1] von das_receive_dttm ->
root/header/document/origination_dttm

following-sibling::*[1] von das_receive_dttm ->
root/header/document/modification_dttm
```

Der Spezifikationsname muss aus `/root/header/document/software/specification/@V` extrahiert werden. Im Spezifikationsnamen ist das Spezifikationsjahr enthalten. Eine DAS validiert mit der höchsten Version aus dem ermittelten Spezifikationsjahr. Gibt es bspw. für das Spezifikationsjahr 2022 die Versionen V01, V02 und V03, so wird die Exportdatei mit der Version V03 validiert.

Der Datenservice für ein Spezifikationsjahr kann am 31. Dez des Folgejahres deaktiviert werden. Exportdateien bspw. mit dem Wert "2025 QSFFx V02" in `/root/header[1]/document[1]/software[1]/specification[1]/@V` müssen also bis zum 31.12.2026 angenommen und verarbeitet werden.

## 7.1 Verarbeitungsschritte ohne Verwendung der Spezifikationskomponente Datenprüfprogramm

Die Abfrage QSDOK.vPruefung (siehe Tabelle 37) zeigt die notwendigen Prüfungen in einer Datenannahmestelle. Verletzte Prüfungen sind schemakonform zu `datenflussprotokoll.xsd` zu protokollieren.



### Achtung bei der Verwendung der XSL-Stylesheets ohne DPP

Im Zusammenhang mit den Prüfungen 205 bis 238 müssen DAS ab der Spezifikation 2025 Werte für die Parameter `dokumentationsanlass_vorgaenger` und `dokumentationsanlass_nachfolger` dem XSL-Stylesheet übergeben (siehe QSDOK.vPruefung.regelverletzung).

Die Metadaten (`case/case_admin`) eines Datensatzes werden grundsätzlich immer persistiert. Das Attribut `meta_DS` in QSDOK.vPruefung zeigt, ob die Verletzung einer Prüfung das Persistieren der Metadaten eines Datensatzes verhindert. Wenn eine verletzte Prüfung die Eigenschaft `meta_DS = true` hat, dann werden die Metadaten nicht persistiert.

`/root/header/document/id/@V` muss immer persistiert werden, es sei denn, es liegt eine Schemaverletzung oder Nicht-Wohlgeformtheit der Exportdatei vor. `//document/id/@V` ist dauerhaft die eindeutige ID der Exportdatei **und** des zugehörigen Datenflussprotokolls. Der Wert wird ausschließlich vom Leistungserbringer geschrieben. `//document/id/@V` muss eine GUID

oder eine UUID sein. Zwei verschiedene Exportdateien werden durch zwei unterschiedliche `//document/id/@V` identifiziert.

Die vom Leistungserbringer geschriebene `//document/id/@V` wird für das Datenflussprotokoll in `//document/id/@V` verwendet. Datenflussprotokolle sind ein Pipeline-Output der Exportdateien. Ein Datenflussprotokoll ist eine gelieferte Exportdatei mit protokollierten Prüfungen und herausgeschnittenem `qs_data`. Bis auf

```
/root/body/data_container/cases/case/qs_data
/root/header/document/modification_dttm/@V
/root/header/provider
/root/header/protocol/status_document/@V
//case/case_admin/protocol/status_case/@V
```

werden die vom Leistungserbringer geschriebenen Werte im Datenflussprotokoll nicht geändert.

Sollte die Exportdatei nicht wohlgeformt sein oder das entsprechende Schema verletzen, dann ist ein Miniprotokoll zu erstellen und kein vollständiges Datenflussprotokoll. Ein Miniprotokoll ist im Unterschied zu Datenflussprotokollen ohne `root/header` und `root/body`. In den XML-Beispielen ist ein valides Miniprotokoll mit dem optionalen XML-Attribut `xsi:schemaLocation` und dem Wert `urn:gba:sqg ../datenflussprotokoll.xsd` enthalten. Bei Schemaverletzungen oder Nicht-Wohlgeformtheit ist eine deutschsprachige Fehlermeldung des genutzten XML-Prozessors in

```
/root/protocol/validation_provider/validation_item/status/error/error_message/@V
```

ausreichend.

Sind Wohlgeformtheit und Schemavalidität gegeben, kann ein vollständiges Datenflussprotokoll erstellt werden. Dazu werden die weiteren Prüfungen aus `QSDOK.vPruefung` geprüft.

Ist eine Prüfung mit der Eigenschaft `vPruefung.protokoll-Level == "DK"` verletzt, dann muss diese Prüfung auf Dokumentenebene (DK) unter

```
/root/header/protocol/validation_provider/validation_item/status/error
```

protokolliert werden.

`QSDOK.vPruefung.bereich ->`

```
/root/header/protocol/validation_provider/validation_item/@V
```

`QSDOK.vPruefung.fkFehlermeldung -->`

```
/root/header/protocol/validation_provider/validation_item/status/error/rule_id/@V
```

`QSDOK.vPruefung.Streng`: ERROR entspricht H, WARNING entspricht W -->

```
/root/header/protocol/validation_provider/validation_item/status/error/rule_type/@V
```

QSDOK.vPruefung.fehlermeldung -->

```
/root/header/protocol/validation_provider/validation_item/status/error/error_message/@V
```

Werden für dieses validation\_item bzw. Prüfbereich auch Prüfungen (oder QSDOK.Regeln) auf Datensatzebene protokolliert, dann ist die Fehlermeldung selbst unter

```
/root/body/data_container/cases/case/case_admin/protocol
```

zu schreiben. Die Zuordnung zwischen

```
/root/header/protocol/validation_provider/validation_item
```

und

```
//case/case_admin/protocol/validation_item
```

erfolgt über die

```
validation_item/@id.
```

Es kann kein validation\_item unter case/case\_admin/protocol ohne das zugehörige validation\_item im header geben.

Im Unterschied zu PPP setzt in QSFFx jeder Fehler mit

```
rule_type = "H" den Wert in //status_document/@V
```

auf ERROR.

In PPP sind anders als in QSFFx mehrere Datensätze in einer Exportdatei möglich. Erst wenn alle Datensätze fehlerhaft ("H") sind, muss

```
//status_document/@V
```

dem Wert ERROR entsprechen.

**Ein Fehler mit rule\_type = "H" auf Datensatzebene hat die Kaskade:**

setze

```
./ancestor::status/@V := "ERROR";
```

```
./ancestor::protocol/status_case/@V := "ERROR";
```

wenn alle Datensätze in der Exportdatei den status\_case/@V := "ERROR" haben,

dann {

unter //header/protocol im validation\_item mit derselben @id:

```
/status/@V := "ERROR";
```

```
/root/header/protocol/status_document/@V := "ERROR";
```



```
}ansonsten {  
  
unter //header/protocol im validation_item mit derselben @id:  
    /status/@V := "WARNING";  
/root/header/protocol/status_document/@V := "WARNING";  
}
```

**Ein Fehler mit rule\_type = "H" auf Dokumentenebene hat die Kaskade (nur im header):**

```
./ancestor::status/@V := "ERROR";  
/root/header/protocol/status_document/@V := "ERROR";
```

**Ein Fehler mit rule\_type = "W" auf Datensatzebene hat die Kaskade:**

```
wenn ./ancestor::status/@V = "OK", dann ./ancestor::status/@V := "WARNING";  
  
wenn ./ancestor::protocol/status_case = "OK", dann  
    ./ancestor::protocol/status_case/@V := "WARNING";  
  
wenn im //header//validation_item mit derselben validation_item/@id vom Daten-  
satz und das validation_item im header hat status/@V = "OK", dann  
    /root/header/protocol/validation_provider/validation_item/status/@V :=  
    "WARNING";  
  
wenn //status_document/@V = "OK", dann  
    /root/header/protocol/status_document/@V := "WARNING";
```

**Ein Fehler mit rule\_type = "W" auf Dokumentenebene (nur im header):**

```
wenn ./ancestor::status/@V = "OK", dann ./ancestor::status/@V := "WARN-  
ING";  
  
wenn //status_document/@V = "OK", dann  
    /root/header/protocol/status_document/@V := "WARNING";
```

Die jeweiligen Zustände können auch berechnet werden.

Ermittlung von /case/case\_admin/protocol/validation\_item/status/@V:

```
if(status//error/rule_type/@V = "H"){  
    status/@V := "ERROR"  
}  
else if(status//error/rule_type/@V = "W"){  
    status/@V := "WARNING"  
}  
else{  
    status/@V := "OK"
```

```
}
```

Ermittlung von /case/case\_admin/protocol/status\_case/@V:

```
if(/case_admin/protocol/validation_item/status/@V = "ERROR") {
    case/case_admin/protocol/status_case/@V := "ERROR"
}else if(/case_admin/protocol/validation_item/status/@V = "WARNING") {
    case/case_admin/protocol/status_case/@V := "WARNING"
}else{
    case/case_admin/protocol/status_case/@V := "OK"
}
```

Ermittlung von

/root/header/protocol/validation\_provider/validation\_item/status/@V mit validation\_item/@id = "y"

```
if(./validation_item/status/error/rule_type/@V = "H" ||
not(//case/case_admin/protocol/validation_item[./@id = "y"]/status/@V =
"OK" || //case/case_admin/protocol/validation_item[./@id = "y"]/sta-
tus/@V = "WARNING")){
    ./validation_item/status/@V := "ERROR"
}else if(./validation_item/status/error/rule_type/@V = "W" ||
//case/case_admin/protocol/validation_item[./@id = "y"]/status/@V =
"WARNING"){
    ./validation_item/status/@V := "WARNING"
}else{
    ./validation_item/status/@V := "OK"
}
```

Ermittlung von /root/header/protocol/status\_document/@V:

```
if(/root/header/protocol/validation_provider/validation_item/status/@V
= "ERROR" || not(//case/case_admin/protocol/status_case/@V = "OK" ||
//case/case_admin/protocol/status_case/@V = "WARNING")){
    /root/header/protocol/status_document/@V := "ERROR"
}else if(/root/header/protocol/validation_provider/validation_item/sta-
tus/@V = "WARNING"){
    /root/header/protocol/status_document/@V := "WARNING"
}else{
    /root/header/protocol/status_document/@V := "OK"
}
```

QSDOK.Regeln sind dem Prüfbereich Spezifikation (validation\_item/@v) zugeordnet, siehe OSDOK.vPruefung mit idPruefung 51. In einer Datenannahmestelle werden nur Regeln mit QSDOK.Regeln.gueltigNachExport = true geprüft.

Die in einer Regel enthaltenen Bogenfelder führt die Tabelle QSDOK.RegelFelder auf. Die Operatoren sind in QSDOK.SyntaxOperator definiert. Grundsätzlich kann eine Regel auch Schlüssel aus QSDOK.Schluesssel oder eine Funktion aus QSDOK.SyntaxFunktion enthalten.

Das @-Zeichen in der beispielhaften Regel

```
@ANMELDUEBERGANG KEINSIN (1) UND @ANMELDEJ KEINSIN (1)
```

referenziert das genannte Datenfeld in jedem Teildatensatz. Für @ANMELDUEBERGANG hieße das ancestor::bogen//ANMELDUEBERGANG.

Ist eine Regel aus QSDOK.Regeln verletzt, so ist der Fehler auf Datensatzebene zu protokollieren:

"Spezifikation" -->

```
//case/case_admin/protocol/validation_item/@v
```

"REGEL" -->

```
//case/case_admin/protocol/validation_item/status/error/error_type/@v
```

QSDOK.Regeln.idRegeln -->

```
//case/case_admin/protocol/validation_item/status/error/rule_id/@v
```

QSDOK.Regeln.fkRegelTyp: hart=H, weich=W -->

```
//case/case_admin/protocol/validation_item/status/error/rule_type/@v
```

QSDOK.Regeln.meldung -->

```
//case/case_admin/protocol/validation_item/status/error/error_message/@v
```

Im Abschnitt 9.3.10 werden die Voraussetzungen (Vorbedingungen) beschrieben, wann eine Regel zu prüfen ist. Die Formulierungen gelten spezifikationsübergreifend und sind eng mit Softwareanbietern abgestimmt, da das Regelparsing auf diesen Evaluationsbedingungen basiert. Diese Evaluationsbedingungen gelten nicht für QSDOK.vPruefung.

Im Zusammenhang mit den harten Schemata bedeuten die Evaluationsbedingungen bspw. für QSFFx:

- Es wird über jeden Teildatensatz mit QSDOK.Bogen.name = "C" iteriert und **jede Regel aus QSDOK.Regeln mit QSDOK.Regeln.gueltigNachExport = true wird geprüft**. Dieses Vorgehen ist möglich, da es keine Regeln mit Regelfeldern gibt, die ausschließlich auf dem Teildatensatz mit QSDOK.Bogen.name = "B" für das Modul FFX liegen.

Eine QSDOK.Regeln.bedingung wird **nicht** geprüft, wenn bei den Operationen mit

```
<>, <=, <, >=, >, =, *, +, -, /, DIV
```

ein Bogenfeld `null` ist UND im Fall von `=` sowie `<>` keiner der Operanden `LEER` entspricht.

**Beispiel:**

Regel: `A <> 0 ODER B <> 0`

wird geprüft, wenn die Vorbedingung gilt: `A <> LEER UND B <> LEER`

Trotz bedingter Prüfung (short-circuit evaluation) – wenn `A <> 0 = WAHR`, dann ist die Prüfung von `B <> 0` überflüssig – muss die Vorbedingung `A <> LEER UND B <> LEER` für `A <> 0 ODER B <> 0` laut Evaluationsbedingungen gelten.

**Beispiel:**

Regel: `C = 1 UND B = LEER`

wird geprüft, wenn die Vorbedingung gilt: `C <> LEER`

**Beispiel:**

Regel: `(C = 1 UND B <> LEER) ODER B > 1`

wird geprüft, wenn die Vorbedingung gilt: `C <> LEER UND B <> LEER`

**Beispiel:**

Regel: `@ANMELDUEBERGANG KEINSIN (1) UND @ANMELDEJ KEINSIN (1)`

hat keine Vorbedingung (da `KEINSIN <>` Vergleichsoperator)

Sollte ein Operand `x` vor einer Operation mit

`<>, <=, <, >=, >, =, *, +, -, /, DIV`

gegen `LEER` mit `"=`" geprüft werden und beide Operationen sind mit einem `"ODER"` verbunden und geklammert, wird die Regel auch geprüft, wenn `x null` ist. Dieser vorherige Satz muss genau der folgenden Syntax entsprechen:

`"(Bogenfeld = LEER ODER Bogenfeld Operator NichtBogenfeld)"`

**Beispiel:**

Regel: `(A = LEER ODER A <> 1) UND (B = LEER ODER B <> 1) UND C <> LEER`

hat keine Vorbedingung

**Beispiel:**

Regel: `(A = LEER ODER A <> 1) UND (B = LEER ODER B <> A) UND C <> LEER`

wird geprüft, wenn die Vorbedingung gilt:

Regel: `A <> LEER UND B <> LEER (aufgrund von B <> A)`

### Beispiel:

Regel: NICHT(A <> LEER UND A = 1) UND (B = LEER ODER B <> 1) UND C <> LEER

wird geprüft, wenn die Vorbedingung gilt: A <> LEER (aufgrund von "<>" sowie "UND" in "NICHT(A <> LEER UND A = 1)", trotz der de-morganschen Gesetze)

Die erstellten Datenflussprotokolle müssen valide gegen datenflussprotokoll.xsd sein.

## 7.2 Verarbeitungsschritte bei Anwendung der Spezifikationskomponente Datenprüfprogramm

Die Verwendung der Spezifikationskomponente Datenprüfprogramm ist nicht verpflichtend.

In Verbindung mit der bereitgestellten Spezifikationskomponente Datenprüfprogramm lassen sich die Verarbeitungsschritte in einer Datenannahmestelle chronologisch untergliedern in:

- Wohlgeformtheit und Schemavalidierung
  - Ausführung des Datenprüfprogramms mit `--no-spez-val`
- Schreiben des XML-Elementes `das_receive_dttm` (Zeitstempel des Dateneingangs) in der geprüften Exportdatei aus `.\output\files\`
- Ausführen der Prüfungen aus QSDOK.vPruefung mit `dpp = false`
- Ausführung des Datenprüfprogramms mit `--no-schema-val` (ohne Schemavalidierung)
- Versand des Datenflussprotokolls an den Leistungserbringer
- Hinweis: Persistieren der Metadaten aus `case/case_admin`

Das Datenprüfprogramm prüft:

- Wohlgeformtheit und Schemakonformität der Exportdatei
- Regeln aus der Tabelle QSDOK.Regeln mit `gueltigNachExport = true`
- administrative Prüfungen aus QSDOK.vPruefung mit der Eigenschaft `dpp = true`

### Wohlgeformtheit und Schemavalidierung

Um die Exportdatei auf Wohlgeformtheit und Schemavalidität zu prüfen, wird das Datenprüfprogramm mit dem Argument `--no-spez-val` aufgerufen:

```
java -jar datenpruefprogramm-4.3.2-jar-with-dependencies.jar -c
config_LE_und_LVKK_data_flow_LE_LVKK.xml --input
.\input\exportdatei.xml --output output --xsd-path
.\xsd\LE_und_LVKK_data_flow_LE_LVKK.xsd --xsl-path
.\xsl\2024_QSFFx_DPP_V01.xsl --no-spez-val
```

Die nächste Zeile gehört nicht zum DPP Aufruf. Verdeutlicht aber, wann eine Exportdatei nicht gegen das Schema valide ist.

```
rem not exist .\output\files\exportdatei.xml -AND exist
.\output\protocol\exportdatei.xml
```

```
if not exist .\output\files\exportdatei.xml if exist
.\output\protocol\exportdatei.xml echo "Die gelieferte
Exportdatei ist nicht valide. Daher wurde ein Miniprotokoll
erzeugt."
```

Wenn die Exportdatei nicht gegen LE\_und\_LVKK\_data\_flow\_LE\_LVKK.xsd valide ist, dann:

- Validierung wird beendet.
- Miniprotokoll aus .\output\protocol\exportdatei.xml an den LE verschicken.
- Der Datensatz (//case\_admin und //qs\_data) aus der gelieferten Exportdatei wird nicht persistiert.

Sind Wohlgeformtheit und Schemakonformität nicht verletzt, kann der **Zeitstempel des Dateneingangs** in der geprüften Datei

```
.\output\files\exportdatei.xml
```

geschrieben werden. Es wird weitergearbeitet mit der geprüften Datei aus .\output\files\exportdatei.xml, denn hier wurde bereits der erste Prüfschritt protokolliert.

Position von root/header/document/das\_receive\_dttm/@V :

```
preceding-sibling::*[1] von das_receive_dttm ->
root/header/document/origination_dttm

following-sibling::*[1] von das_receive_dttm ->
root/header/document/modification_dttm
```

Das DPP unterstützt **nicht** das Schreiben von:

```
/root/header/document/das_receive_dttm/@V.
```

### Ausführen der Prüfungen aus QSDOK.vPruefung mit dpp = false

Wenn die Dateien .\output\files\exportdatei.xml UND .\output\protocol\exportdatei.xml existieren, dann werden die Prüfungen protokolliert, die nicht Built-In Prüfungen des Datenprüfprogramms sein können:

siehe QSDOK.vPruefungen mit DPP = false

Diese Prüfungen sind entweder auf Dokumenten- oder auf Datensebene zu protokollieren (siehe Abschnitt 7.1 Verarbeitungsschritte ohne Verwendung der Spezifikationskomponente Datenprüfprogramm).

Sind alle Prüfungen aus QSDOK.vPruefung mit dpp = false abgeschlossen und //status\_document/@V entspricht dem Wert "ERROR":

- Die Validierung wird beendet.
- Das Datenflussprotokoll .\output\protocol\exportdatei.xml an den LE verschicken.
- //qs\_data wird nicht persistiert.

## Ausführung des Datenprüfprogramms mit `--no-schema-val` (ohne Schemavalidierung)

In diesem Schritt werden die folgenden Punkte durch das Datenprüfprogramm geprüft:

- Regeln aus der Tabelle QSDOK.Regeln mit `gueltigNachExport = true`
- administrative Prüfungen aus QSDOK.vPruefung mit der Eigenschaft `dpp = true`

exemplarischer Aufruf des Datenprüfprogramms mit dem Einzeiler:

```
java -jar datenpruefprogramm-4.3.2-jar-with-dependencies.jar -c
config_LE_und_LVKK_data_flow_LE_LVKK.xml --input
.\input\exportdatei.xml --output output --xsd-path
.\xsd\LE_und_LVKK_data_flow_LE_LVKK.xsd --xsl-path
.\xsl\2024_QSFFx_DPP_V01.xsl --no-schema-val
```

Wenn im Ergebnis `//status_document/@V` ungleich "ERROR" ist, dann werden die Kind-Element von `qs_data` aus `.\output\file\exportdatei.xml` persistiert. Wenn `status_document` gleich "ERROR" ist, dann bleiben bereits persistierte Kind-Element von `qs_data` aus einem bereits gelieferten, validen Datensatz bestehen.

Final wird dann das Datenflussprotokoll `.\output\protocol\exportdatei.xml` an den LE verschickt.



### **Persistieren von `case_admin` auch bei ERROR in `status_document/@V` oder `status_case/@V`**

Die Metadaten aus `case_admin` werden grundsätzlich immer persistiert. Sollte eine verletzte Prüfung die Eigenschaft `meta_DS = true` haben, dann werden die Metadaten nicht persistiert. Sind nur Prüfungen mit `meta_DS = false` verletzt, müssen die Metadaten aus `case/case_admin` unabhängig von `status_document` und `status_case` in der DAS persistiert werden.

## 7.3 Hinweise

- Aufgrund von Race Conditions und aus anderen Gründen könnte es selbst in einem direkten Datenfluss möglich sein, dass ein `update` eher die Datenannahmestelle erreicht als ein `create`. Daher sind `create` und `update` in einer Datenannahmestelle gleich zu behandeln. Entscheidend ist `case_admin/version/@V`. Sollte ein `update` mit Version 2 bereits in der Datenannahmestelle persistiert sein, dann ist ein später eintreffendes `create` mit `case_admin/version/@V` gleich 1 abzulehnen (siehe auch QSDOK.vPruefung mit `idPruefung` 37 und 48).
- Datensätze mit `case_admin/action/@V = "delete"` müssen nicht `qs_data` enthalten.
- Sollte nach einem `delete` ein valider Datensatz mit `delete`, `create` oder `update` geliefert werden – valide auch bzgl. Prüfung 37 – dann ist dieser Datensatz zu persistieren, sprich, dieser Datensatz überschreibt `delete`.
- Bzgl. QSDOK:

- alle Attribute, die Schlüssel anderer Tabellen referenzieren, beginnen immer mit dem Präfix "fk". Beispiel: Attribut fkModul in der Tabelle QSDOK.Regeln
- Die Tabelle QSDOK.ExportFormatExportModul zeigt alle möglichen Exportfelder für einen Datensatz (case). Lesebeispiel aus der QSFFx\_QSDOK: Das Exportfeld ANMELDEJ ist maximal einmal (Attribut maxAnzahl) auf dem Bogen (Teildatensatz/Objekt) mit dem Namen C. Mutterbogen B hat mindestens einen Kind-Bogen C. Das Exportmodul F\_NW hat genau einen Bogen B (Basisbogen). Exportfeld ANMELDEJ ist vom BasisTyp NUMSCHLUESSEL (Schlüssel mit ganzzahligen Schlüsselcodes) und ist einstellig (Attribut maxZeichenlaenge).

Tabelle 7: Struktur der Tabelle *ExportFormatExportModul*

Feldname	Bemerkung
idExportFormatExportModul	Primärschlüssel
Exportmodul_name	name aus Tabelle Exportmodul
fkExportmodul	Fremdschlüssel zur Tabelle Exportmodul
qs_data_xsi_type	type_QS_data oder type_QS_data_mds aus Tabelle Exportmodul (Wenn Wert null, dann führt xpath zu keinem Kind-Element von qs_data.)
Bogen_name	name aus Tabelle Bogen (Wenn Wert null, dann führt xpath zu keinem Kind-Element von qs_data.)
fkBogen	Fremdschlüssel zur Tabelle Bogen
BogenZahl_bzgl_mutterbogen	Anzahl der Teildatensätze/Bögen bezogen auf den Mutterbogen (QSDOK.Bogen.fkBogenZahl)
Bogen_mutterbogen_name	name aus Tabelle Bogen
fkMutterbogen	Schlüssel zum Mutterbogen
exportfeld	name entweder aus Tabelle Zusatzfeld, Ersatzfeld oder Feld
lfdNr	Sortierhilfe innerhalb eines Bogens; untere Grenze kann größer als 1 sein (lfdNr 4 kann bspw. dem xpath child::*[1] entsprechen); Wenn Wert gleich 0, dann führt xpath zu keinem Kind-Element von qs_data.
maxAnzahl	maximale Anzahl des Exportfeldes auf dem Bogen (Wenn fkBogen gleich null, dann maximale Anzahl im Datensatz.)
BasisTyp_name	name aus Tabelle BasisTyp



Feldname	Bemerkung
fkBasisTyp	Fremdschlüssel zur Tabelle BasisTyp
maxZeichenlaenge	maximale Zeichenlänge
xpath	xpath innerhalb der Exportdatei (Ausgangspunkt bzw. current node ist <case>)
fkMussKann	fkMussKann aus Tabelle MussKann
fkBogenFeld	Fremdschlüssel zur Tabelle BogenFeld
BogenFeld_name	name aus Tabelle BogenFeld
BogenFeld_bezeichnung	bezeichnung aus Tabelle BogenFeld
fkErsatzFeld	Fremdschlüssel zur Tabelle ErsatzFeld
ErsatzFeld_name	name aus Tabelle ErsatzFeld
ErsatzFeld_bezeichnung	bezeichnung aus Tabelle ErsatzFeld
fkZusatzFeld	Fremdschlüssel zur Tabelle ZusatzFeld
ZusatzFeld_name	name aus Tabelle ZusatzFeld
ZusatzFeld_bezeichnng	bezeichnung aus Tabelle ZusatzFeld
fkModul	Fremdschlüssel zur Tabelle Modul
Modul_name	name aus Tabelle Modul
fkExportZiele	Fremdschlüssel zur Tabelle ExportZiele
fkExportZielXml	idExportZielXml aus Tabelle ExportZielXml
Bogen_sortierNr	sortierNr aus Tabelle Bogen
fkExportformat	idExportformat aus Tabelle Exportformat

## B Komponenten

In diesem Abschnitt werden die einzelnen Komponenten der Spezifikation beschrieben, die in ihrer Gesamtheit ein Spezifikationspaket abbilden.

## 8 Spezifikationskomponenten

Die Spezifikationskomponenten für die QS-Dokumentation dienen der Erstellung von Software zur Datenerfassung, Plausibilitätsprüfung und Datenübermittlung für die externe vergleichende Qualitätssicherung im Rahmen der Strukturabfragen gemäß den Richtlinien QFR, PPP und QSFFx. Diese sollen die Bereitstellung valider und vergleichbarer Daten gewährleisten.

### 8.1 Spezifikationskomponenten

Ein Spezifikationspaket kann sich aus den folgenden Spezifikationskomponenten zusammensetzen:

#### 8.1.1 Datenbank

Die Datenbank bezeichnet die Access-Datenbank, in der die QS-Dokumentation spezifiziert wird. Sie dient der (automatisierten) Erstellung von Software für QS-Dokumentation. Folgende Spezifikationskomponente wird als Access-Datenbanken (MS Access 2007-2013) zur Verfügung gestellt:

- Datenbank zur QS-Dokumentation (QSDOK) – Die Datenbank zur QS-Dokumentation dient der Spezifikation von Datenerhebung und -erfassung unter Berücksichtigung von Plausibilitätsprüfungen und zu exportierenden Datenfeldern.

Weiterführende Angaben zur QSDOK sind dem Kapitel 9 zu entnehmen.

#### 8.1.2 Schema

Der Ordner Schema auf der Komponentenebene ist eine ZIP-Datei, die die Versionierung und vollständige Bezeichnung enthält. Sie enthält einzelne XML-Schema, die festlegen, in welcher Struktur XML-Daten an Schnittstellen im Datenfluss vorliegen müssen.

#### 8.1.3 Ausfüllhinweise

Der Ordner Ausfüllhinweise auf der Komponentenebene ist eine ZIP-Datei, welche die Versionierung und vollständige Bezeichnung enthält. Sie enthält einzelne HTML-Dateien für jedes Modul, die mit den Kürzeln der einzelnen Module benannt sind. Die Ausfüllhinweise (Ausfüllhinweise) dienen als Hilfestellung bei der Dokumentation durch den Anwender. Die Namen der HTML-Dateien für einzelne Datenfelder sind in der Datenbank (`BogenFeld.ahinweis`) hinterlegt.

#### 8.1.4 Dokubögen

Der Ordner Dokubögen auf der Komponentenebene ist eine ZIP-Datei, die die Versionierung und vollständige Bezeichnung enthält. Sie beinhaltet die Dokumentationsbögen als einzelne PDF-Dateien für jedes Modul, die mit den Kürzeln der einzelnen Module benannt sind. Die Dokumentationsbögen bilden einige wichtige Datenbankinhalte<sup>14</sup> ab.

#### 8.1.5 TechDok

Die TechDok bezeichnet alle technischen Dokumentationen, die detaillierte Erläuterungen zur Funktionsweise und Verwendung der einzelnen Komponenten geben. Die Technische Dokumentation beschreibt hier die Spezifikation für Verfahren der Strukturabfrage. Wie in der Einleitung schon beschrieben, richtet sie sich dabei an Orte der Leistungserbringung (wie zum Beispiel Krankenhäuser und andere medizinische Einrichtungen), die als Leistungserbringer (LE) bezeichnet werden. Folgende Dokumente zählen zu der Spezifikationskomponente TechDok:

- Allgemeine Technische Dokumentation (allgemein gültige Vorgaben für QFR-RL, PPP-RL und QSFFx-RL)
- Verfahrensspezifische Technische Dokumentation für die Spezifikation zur Strukturabfrage nach QSFFx-RL
- Verfahrensspezifische Technische Dokumentation für die Spezifikation zur Strukturabfrage nach PPP-RL
- Verfahrensspezifische Technische Dokumentation für die Spezifikation zur Strukturabfrage nach QFR-RL
- Technische Dokumentation für Benennungsschema von Spezifikationen

#### 8.1.6 ÜbersichtÄnderungen

Die Datei UebersichtAenderungen ist eine PDF-Datei mit der Übersicht über die Änderungen in der Spezifikation zu allen Vorversionen des Spezifikationsjahres und zur letzten Version des Vorjahres. Hier werden die Änderungen übersichtlich zusammengefasst und erläutert.

#### 8.1.7 Komponentenübersicht

Die Komponentenuübersicht bezeichnet die Auflistung aller im Spezifikationspaket enthaltenen Komponenten mit Bezug zu Versionsnummer und Veröffentlichungsdatum (CSV).

#### 8.1.8 Datenprüfprogramm

Der Ordner Datenpruefprogramm ist auf Komponentenebene eine ZIP-Datei, welche die Bestandteile des Datenprüfprogramms (DPP) enthält. Hilfsprogramme, wie etwa das Datenprüfprogramm, werden ggf. ebenfalls als Komponenten in ein Spezifikationspaket aufgenommen. Bei der

---

<sup>14</sup> Die Papierform ist nur als eine Abbildung des Eingabeformulars oder der Eingabemaske zu verstehen und darf nicht zur tatsächlichen QS-Dokumentation genutzt werden. Verbindlich sind daher nur die Inhalte der Datenbank zur QS-Dokumentation.

Angabe der Betriebsart und des Exportformats gelten die gleichen Abkürzungen wie bei den Spezifikationspaketen. Diese Angaben erfolgen aber nur dann, wenn sich die Komponenten durch diese Merkmale unterscheiden.



#### **Hinweis**

V<Versionsnummer>: Die Versionierung der Komponenten erfolgt in ganzen Zahlen, die zweistellig angegeben sind (unter 10 mit einer vorstehenden 0, z. B. V01).

---

## **8.2 Weitere spezifikationsrelevante Dateien und Programme**

Zusätzlich zu den soeben aufgeführten Spezifikationskomponenten gibt es weitere Bestandteile, die nicht zu den Spezifikationskomponenten gehören:

### **8.2.1 Verschlüsselungsprogramm**

Die Verschlüsselungsprogramme (Verschlüsselungsprogramme) werden nicht als Spezifikationskomponente veröffentlicht. Sie sind daher keine dem Spezifikationspaket zugehörige Komponente, verfügen jedoch über dieselbe Verbindlichkeit. Da sie ein eigenständiges Paket in Form einer ZIP-Datei darstellen, können sie außerhalb des Releasezyklus angepasst werden. In der ZIP-Datei enthalten sind der X- und der TPACKER sowie die zum Veröffentlichungszeitpunkt aktuellen öffentlichen Schlüssel der Datenservices im Datenfluss.

### **8.2.2 Datenbank zu Datenserviceinformationen**

Die Datenserviceinformationen (Abschnitt 9.7.2) werden in einer separaten Datenbank gepflegt. Die Datenbank zu Datenserviceinformationen ist keine dem Spezifikationspaket zugehörige Komponente, verfügt jedoch über dieselbe Verbindlichkeit. Da sie eine eigenständige Spezifikationsdatenbank darstellt kann sie außerhalb des Releasezyklus angepasst werden.

### **8.2.3 Excel-Datei zur optionalen Anwendung beim Verfahren nach PPP-RL**

Die Verwendung der Excel-Abfrage-Datei ist optional und obliegt der jeweiligen Institution (z. B. den Landesaufsichtsbehörden oder den Medizinischen Diensten).

Grundsätzlich ist die Excel-Datei so aufgebaut, dass im Hintergrund eine Ordner-Abfrage (Power-Query) durchgeführt wird und XML-Dateien ausgewertet werden können, die sich in einem auszuwählenden Ordner befinden. Erklärungen und Hinweise, wie der gewünschte Ordner gesetzt und die Excel-Abfrage aktualisiert werden kann, ist dem Tabellenreiter „Anleitung“ zu entnehmen.

## 9 QS-Dokumentation

Die vorliegenden Spezifikationskomponenten für die QS-Dokumentation dienen der Erstellung von Software zur Datenerfassung, Plausibilitätsprüfung und Datenübermittlung für die externe vergleichende Qualitätssicherung im Rahmen der Strukturabfragen gemäß den Richtlinien QFR, PPP und QSFFx. Diese sollen die Bereitstellung valider und vergleichbarer Daten gewährleisten. Wie unter B Komponenten auf Seite 70 beschrieben, zählen neben der Datenbank zur QS-Dokumentation unter anderem die Ausfüllhinweise und die Dokumentationsbögen zu den Spezifikationskomponenten. Die Delta-Tabellen in der Datenbank zur QS-Dokumentation stellen geänderte, gelöschte und neue Datenbankinhalte im Vergleich zur letzten gültigen Version des Vorjahres oder zur letzten Version des aktuellen Spezifikationsjahres dar.

### 9.1 Anmerkungen zur Struktur der Spezifikation zur QS-Dokumentation

Die Spezifikation zur QS-Dokumentation ist in einer relationalen Datenbank abgelegt. Zurzeit wird sie ausschließlich als Access-Datenbank (MS Access 2007-2013) zur Verfügung gestellt. Wie in der Spezifikationskomponente „Technische Dokumentation zum Benennungsschema von Spezifikationen“ beschrieben, hat der Name der Spezifikation folgendes Schema:

`<Spezifikationsjahr>_<Richtlinie>_QSDOK_V<Versionsnummer>.mdb`

Der Wert `<Spezifikationsjahr>` bezeichnet das Jahr, in dem die QS-Dokumentation stattfindet. `<Richtlinie>` bezeichnet die Richtlinie, der die Spezifikation zugrunde liegt (z. B. QSFFx-RL). `<Versionsnummer>` bezeichnet die 2-stellige Versionsnummer (z. B. 01).

---

#### Beispiel:

Im Spezifikationsjahr 2025 ist die Spezifikation in der `2025_PPP_QSDOK_V01.mdb`<sup>15</sup> gültig.

---

#### 9.1.1 Tabellenstruktur der Datenbanken

Die Tabellen und deren Spalten (Attribute) unterliegen einem einheitlichen Namensschema. Erlaubte Zeichen sind die Buchstaben a-z, A-Z und die Ziffern 0-9. Umlaute und Sonderzeichen werden nicht verwendet. Das erste Zeichen eines Namens darf keine Ziffer sein.

Ein Tabellename beginnt immer mit einem Großbuchstaben und ein Attributname mit einem Kleinbuchstaben. Wenn ein Name aus mehreren Teilen (z. B. Substantiven) besteht, so beginnt jeder nachfolgende Namensteil mit einem Großbuchstaben.

---

<sup>15</sup> Die Versionsnummer der gültigen Spezifikation (z. B. V01, V02 usw.) ist dem zuletzt veröffentlichten Update zu entnehmen.

---

**Beispiel:**

BasisTyp (Tabelle)

idBasisTyp (Spalte)

---

Für jede Tabelle ist in der Spezifikation ein Primärschlüssel definiert, der nachfolgendem Schema aufgebaut ist:

`id<TabellenName>`

Der Ausdruck in spitzen Klammern ist ein Platzhalter für den Namen der Tabelle. Die meisten Tabellen haben einen einfachen Primärschlüssel vom Typ `AUTOINCREMENT`. Zusätzlich enthalten derartige Tabellen mindestens ein identifizierendes Attribut<sup>16</sup>, welches durch Setzen eines weiteren, eindeutigen Indexes (bestehend aus einem oder mehreren Attributen) definiert ist.

---

**Beispiele:**Identifizierendes Attribut: Attribut `name` in Tabelle `BasisTyp`Identifizierende Attributkombination: Attribute `code` und `fkSchluessel` in Tabelle `SchluesselWert`

---

Es gibt auch Tabellen, deren einziger eindeutiger Schlüssel der Primärschlüssel ist. Ein Beispiel ist die Tabelle `MussKann` mit dem Primärschlüssel `idMussKann` vom Typ `TEXT (1)` (entspricht `VARCHAR(1)`). Diese Tabellen sind als einfache „Nachschlagtabellen“ zu interpretieren. Im Fall der Tabelle `MussKann` soll im entsprechenden Fremdschlüsselfeld der verknüpften Detailtabelle durch das Datenbankschema gewährleistet werden, dass nur ein `M` oder `K` eingegeben werden darf.

Die Namen von Fremdschlüsseln sind analog zum Namen der Primärschlüssel aufgebaut:

`fk<FremdTabellenName>`

Die Namensgebung von Primär- und Fremdschlüsseln vereinfacht den Aufbau von komplexeren Abfragen, welche sich über mehrere Entitäten erstrecken (Inklusionsverknüpfungen, Joins).

Die Fremdschlüsselattribute (beginnen mit `fk`) wurden als Datenbankattribute zum Nachschlagen eingerichtet. Zum Beispiel wird beim Fremdschlüsselattribut `fkModul` in der Tabelle `Tds` nicht mehr der Primärschlüssel des jeweiligen Moduls, sondern der Name des Moduls angezeigt.

**Hinweis**

Diese Änderung betrifft nur die Anzeige, nicht jedoch die Struktur der Datenbank.

---

---

<sup>16</sup> Oder sie enthalten eine identifizierende Attributkombination, die einen eindeutigen Schlüssel definiert.

Sind zwei Tabellen mehrfach durch Schlüssel-Fremdschlüssel-Beziehungen miteinander verknüpft, so kann der Name eines Fremdschlüssels auch folgendermaßen aufgebaut sein:

`fk<FremdTabelleName><Rolle>`

`<Rolle>` ist der Platzhalter für eine zusätzliche Qualifizierung der Relation.

N-M-Beziehungen werden wie üblich über Verknüpfungstabellen realisiert. In der Spezifikation haben Verknüpfungstabellen gewöhnlich keinen Primärschlüssel<sup>17</sup>, jedoch einen eindeutigen Schlüssel, der über die Fremdschlüsselfelder definiert ist.

Folgende Attribute treten in vielen Tabellen auf:

- `name` ist in der Regel als technischer Name zu verstehen. Zum Beispiel wird `Feld.name` als Variablenname in den Plausibilitätsregeln verwendet.
- `bezeichnung` ist eine kurze Beschreibung. Zum Beispiel ist `TdsFeld.bezeichnung` der Text, welcher ein Feld auf einem Eingabeformular beschreibt.

`bedingung` enthält einen logischen Ausdruck. Ein bekannter Vertreter dieses Attributtyps ist das Attribut `bedingung` in der Tabelle `Regeln`.

Folgende Attribute treten in vielen Tabellen auf:

- `name` ist in der Regel als technischer Name zu verstehen. Zum Beispiel wird `Feld.name` als Variablenname in den Plausibilitätsregeln verwendet.
- `bezeichnung` ist eine kurze Beschreibung. Zum Beispiel ist `BogenFeld.bezeichnung` der Text, welcher ein Feld auf einem Eingabeformular beschreibt.
- `bedingung` enthält einen logischen Ausdruck. Prominentester Vertreter dieses Attributtyps ist das Attribut `bedingung` in der Tabelle `Regeln`.

### 9.1.2 Abfragen der Datenbank

Die Abfragen der Access-Datenbank geben einen vereinfachenden Überblick über die Inhalte der Spezifikation.

#### Datensätze

Diese Abfrage liefert einen Überblick über die in der Spezifikation enthaltenen Module (verpflichtende und freiwillige Module), die der aktuell gültigen Version entsprechen.

#### Datenfeldbeschreibung

Hier sind alle Bogenfelder der spezifizierten Module, sortiert nach Modulname, Bogenname und Zeilennummer der Bogenfelder dargestellt (Abschnitt 9.2).

---

<sup>17</sup> Hier: Primärschlüssel im Sinne der Access-Definition eines Primärschlüssels. Streng genommen wird über die beiden Fremdschlüssel ein neuer Primärschlüssel definiert.



### **DatenfeldbeschreibungFürEinModul**

Wenn man diese Abfrage aufruft, so muss der Modulname (z. B. „FFX“) angegeben werden und man erhält eine entsprechende modulbezogene Auswahl der Datenfeldbeschreibung.

### **Plausibilitätsregeln**

Diese Abfrage enthält alle Plausibilitätsregeln der spezifizierten Module, sortiert nach Modulname und Nummer der Regel (Abschnitt 9.3).

### **PlausibilitätsregelnFürEinModul**

Wenn man diese Abfrage aufruft, so muss der Modulname (z. B. „FFX“) angegeben werden und man erhält eine entsprechende modulbezogene Auswahl der Plausibilitätsregeln.

### **Teildatensätze**

Diese Abfrage liefert einen Überblick über die Teildatensätze und die Regeln für das Anlegen von Teildatensätzen (Abschnitt 9.2.2).

### **Ersatzfelder**

Dies ist eine Auflistung der Ersatzfelder für die Bogenfelder, die modulspezifisch anonymisiert werden müssen (Export von Teildatensätzen).

### **OPSListen**

Diese Abfrage liefert einen Überblick über die Codes der OPS-Listen (Abschnitt 9.4).

### **ICDListen**

Diese Abfrage liefert einen Überblick über die Codes der ICD-Listen.

### **Exportfelder**

Wenn man diese Abfrage aufruft, erhält man eine Übersicht über alle Exportfelder. Exportfelder für Listenfelder werden nicht pro Listenelement, sondern pro Listenfeld dargestellt. Die Anzahl der Elemente ist der Abfrage zu entnehmen (`Exportfelder.elemente`).

### **ExportfelderFürEinModul**

Diese Abfrage zeigt eine Auswahl der Exportfelder eines Moduls (Modulname ist explizit anzugeben). Man erhält eine Übersicht über die zu exportierenden Felder inkl. Zuordnung zum Teildatensatz. Exportfelder für Listenfelder werden nicht pro Listenelement, sondern pro Listenfeld dargestellt. Die Anzahl der Elemente ist der Abfrage zu entnehmen (`ExportfelderFürEinModul.elemente`).

### **Feldgruppen**

Diese Abfrage liefert eine Übersicht über alle Feldgruppen (Abschnitt 9.3.8).

### **FeldgruppenFürEinModul**

Wenn man diese Abfrage aufruft, so muss der Modulname (z. B. „FFX“) angegeben werden und man erhält eine entsprechende modulbezogene Auswahl der Feldgruppen eines Moduls.

### **WertebereicheNumerischerFelder**

Diese Abfrage liefert eine modulübergreifende Anzeige der numerischen Datenfelder (Typ `ZAHL` und `GANZEZAHL`) und ihrer Wertebereiche.

### **WertebereicheNumerischerFelderFürEinModul**

Hier werden die numerischen Datenfelder (Typ `ZAHL` und `GANZEZAHL`) und ihrer Wertebereiche für ein Modul angezeigt. Das Modul muss bei Aufrufen der Abfrage angegeben werden.

### **ÜberschriftenFürEinModul**

Diese Abfrage liefert eine Anzeige der Überschriften für das angegebene Modul. Angegeben werden Start- und Ende-Felder der Überschriften, sowie die Ebene der Überschriften.

### **Schlüsselcodes**

Diese Abfrage zeigt alle Schlüssel und die zugehörigen Schlüsselwerte an.

### **Ausfüllhinweise**

Hier wird die Zuordnung von Ausfüllhinweisen (htm.Dateien) zu den Feldern in den einzelnen Modulen angezeigt.

### **AusfüllhinweiseFürEinModul**

Hier wird die Zuordnung von Ausfüllhinweisen (htm.Dateien) zu den Feldern eines Moduls angezeigt. Das Modul muss bei Aufrufen der Abfrage angegeben werden.

## **9.2 Datenfeldbeschreibung**

Für jedes Modul existiert eine eigene Datenfeldbeschreibung. Sie spezifiziert alle auszufüllenden Datenfelder (Bogenfelder, auch Items genannt) und besteht aus mehreren Tabellen (Abbildung 6), die in den nachfolgenden Abschnitten erläutert werden.

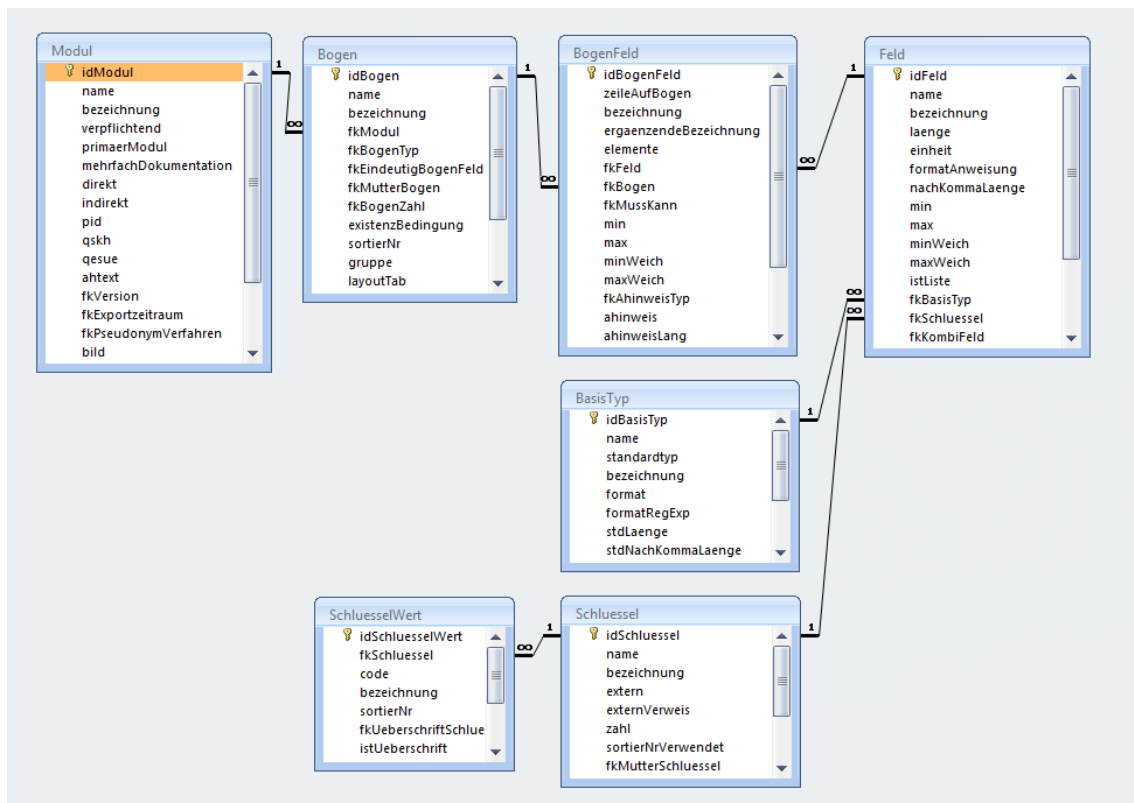


Abbildung 6: Tabellen und Relationen der Datenfeldbeschreibung

Die Abfragen Datenfeldbeschreibung und DatenfeldbeschreibungFürEinModul der Access-Datenbank ermöglichen den Überblick über diese Struktur.

Die Beschreibung der Datenfelder hat folgende Ziele:

- Bereitstellung der Informationen, welche für die Programmierung des Eingabeformulars und für die Sicherung der eingegebenen Daten nötig sind,
- Vermeidung von Redundanzen und
- Typisierung der Felder nach fachlichen und datentechnischen Kriterien

Das für den Anwender wichtigste Merkmal ist die Bezeichnung des Datenfelds (Attribut `BogenFeld.bezeichnung`).

Die Datenfeldbeschreibung ist teilweise auf dem jeweiligen Dokumentationsbogen eines Moduls („Bogensicht“) abgebildet. Zu beachten ist dabei, dass die „Bogensicht“ lediglich die Sicht der medizinischen Fachgruppen, die die Module entwickeln, darstellt. Bei verteilten Softwarelösungen für einen Leistungserbringer hingegen ist die Bogensicht dann nicht mehr adäquat, wenn die Bestandteile eines Bogens auf verschiedene Teilsysteme verteilt sind. Die Daten eines Bogens werden in diesen Fällen für den Export aus den einzelnen Teilsystemen zusammengestellt.



#### Hinweis

Die Papierbögen werden lediglich zu Illustrationszwecken zur Verfügung gestellt. Sie sind zur Dokumentation nicht zugelassen.

Im Kontext einer integrierten, prozessorientierten Dokumentationssoftware müssen die Teildatensätze nicht direkt in Eingabeformulare umgesetzt werden. Es ist sinnvoller, die Teile eines Dokumentationsbogens zu dem Zeitpunkt und in dem Dokumentationskontext zu erfragen, der sich in den Prozessablauf<sup>18</sup> eines Leistungserbringers einordnet.

### 9.2.1 Dokumentationsmodule (Datensätze)

Die Module der Spezifikation enthalten die Datensatzdefinition der jeweiligen Richtlinie. Abhängig von (inhaltlich oder organisatorisch) abzugrenzenden Bereichen kann eine Richtlinie mehrere Module beinhalten. Fehlerfreie Moduldokumentationen (verkürzt „Module“), die die Basis der Datenauswertungen bilden, werden dem Leistungserbringer von der Datenannahmestelle bestätigt. Aus technischer Sicht ist ein Modul durch einen eindeutigen Namen gekennzeichnet. Es umfasst mindestens einen Teildatensatz. In der Tabelle *Modul* der QS-Spezifikation finden sich die zentralen Definitionen eines Moduls.

Tabelle 8: Struktur der Tabelle *Modul*

Feldname	Datentyp	Bemerkung
idModul	INTEGER	Primärschlüssel
Name	TEXT (32)	Eindeutiger technischer Name
bezeichnung	TEXT (255)	Erläuternde Bezeichnung
verpflichtend	BOOLEAN	Besteht für das Modul eine QS-Dokumentationsverpflichtung?
primaerModul	BOOLEAN	Ist das Modul ein Primärmodul?
mehrfachDokumentation	BOOLEAN	Ist ein mehrfaches Anlegen eines gleichartigen Datensatzes pro Fall zulässig (WAHR/FALSCH)?
ahntext	MEMO	Einleitender Text für den Ausfüllhinweis eines Moduls
fkVersion	INTEGER	Gültige Version des jeweiligen Moduls
fkPseudonymVerfahren	INTEGER	Pseudonymisierung von FU-Verfahren
bild	TEXT (20)	Modulspezifisches Bild

### 9.2.2 Teildatensätze

Die Begriffe „Teildatensatz“ und „Bogen“ werden synonym gebraucht. In den der Illustration dienenden Dokumentationsbögen werden alle Teildatensätze aufgeführt. Dabei erfolgt eine chronologische Anordnung, was dazu führen kann, dass ein Teildatensatz durch einen anderen, hierarchisch untergeordneten Teildatensatz unterbrochen wird. Manche Teildatensätze müssen unter

<sup>18</sup> Zum Beispiel in den Prozessablauf eines Krankenhauses.

bestimmten Umständen mehrfach pro Datensatz ausgefüllt werden. Diese mehrfach dokumentierbaren Teildatensätze sind im Muster-Dokumentationsbogen mit entsprechendem Hinweis nur einmal abgebildet.

Ein Teildatensatz

- ist jeweils einem Modul zugeordnet,
- besitzt einen Namen, der innerhalb eines Moduls eindeutig ist,
- kann unter definierten Bedingungen mehrfach pro Fall erzeugt werden.

Die Teildatensätze der QS-Spezifikation sind in der Tabelle *Bogen* definiert (Tabelle 9).

Tabelle 9: Struktur der Tabelle *Bogen*

Feldname	Datentyp	Bemerkung
idBogen	INTEGER	Primärschlüssel
Name	TEXT	Technischer Name des Teildatensatzes
bezeichnung	TEXT	Beschreibender Text
existenzBedingung	MEMO	Logische Bedingung (Regeln für das Anlegen von Teildatensätzen)
sortierNr	INTEGER	Reihenfolge der Unterbögen bei der Erfassung und beim Export
fkModul	INTEGER	Obligatorischer Fremdschlüssel zu einem Modul
fkBogenZahl	TEXT (1)	Anzahl der auszufüllenden Teildatensätze bezogen auf den Basisbogen oder ggf. auf den Mutterteildatensatz
fkMutterBogen	INTEGER	Optionalen Fremdschlüssel, welcher den Mutterteildatensatz eines Teildatensatzes definiert
fkBogenTyp	TEXT (1)	Spezifiziert den für den Export relevanten Bogentyp: Mögliche Werte B, K oder O. Die Angabe ist obligatorisch.
fkEindeutigBogen-Feld	INTEGER	Fremdschlüssel auf ein Bogenfeld, das mehrfach vorhandene Teildatensätze eines Datensatzes identifiziert

### Benennung von Teildatensätzen

Ein Teildatensatz wird durch die folgende Kombination von Modulnamen und Bogenname identifiziert und angesprochen:

<Modul.name>:<Bogen.name>

### Bogentyp

Der Kerndatensatz besteht aus mindestens einem Basisteildatensatz und kann durch einen oder mehrere Teildatensätze ergänzt werden. Das Attribut *Bogen.fkBogenTyp* definiert für jeden

Teildatensatz seine Rolle im und seine Zugehörigkeit zum Kerndatensatz. In Tabelle 10 sind die Bezeichnungen der einzelnen Bogentypen definiert.

Tabelle 10: Inhalte der Tabelle *BogenTyp*

idBogenTyp	Bezeichnung
B	Basisteildatensatz (Teil des Kerndatensatzes)
K	Teildatensatz ist Teil des Kerndatensatzes und kein Basisteildatensatzes
O	Teildatensatz ist Teil des optionalen Datensatzes

### Hierarchie von Teildatensätzen

Der Ausgangspunkt („root“) für die Teildatensatzhierarchie eines Moduls ist immer der Basisteildatensatz (Wert B des Attributs *Bogen.fkBogenTyp*). Ein abhängiger Teildatensatz besitzt einen Mutterteildatensatz, der über das Attribut *fkMutterBogen* der Tabelle *Bogen* definiert ist.<sup>19</sup>

Auf diese Weise lässt sich für jedes Modul ein „Hierarchiebaum“ der Teildatensätze aufbauen.

### Regeln für das Anlegen von Teildatensätzen

Jedes Modul muss die Definition genau eines Basisteildatensatzes enthalten (Wert B des Attributs *fkBogenTyp* der Tabelle *Bogen*). Wenn die Dokumentation eines Moduls durchgeführt wird, muss der Basisteildatensatz genau einmal angelegt werden (z. B. in der Exportdatei). Das Attribut *fkBogenZahl* gibt Auskunft darüber, wie oft ein Teildatensatz pro Vorgang angelegt werden darf. Folgende Werte des Attributs sind möglich:

- 1 = Genau ein Teildatensatz muss ausgefüllt werden
- + = Mindestens ein Teildatensatz muss ausgefüllt werden
- ? = Höchstens ein Teildatensatz darf ausgefüllt werden
- \* = Eine beliebige Anzahl von Teildatensätzen kann ausgefüllt werden

Die Kardinalität eines abhängigen Teildatensatzes bezieht sich auf den Mutterteildatensatz. Der Basisteildatensatz hat immer die Kardinalität 1.

Die Ausprägung *fkBogenZahl* = \* definiert eine 1-n-Beziehung. Zu beachten ist, dass das Attribut *fkBogenZahl* wichtig für den XML-Aufbau des QS-Datensatzes ist und im Schema Berücksichtigung findet.

Weiterhin ist zu beachten, dass die im Attribut *fkBogenZahl* der Tabelle *Bogen* definierten Kardinalitäten durch Definitionen in den nachfolgend beschriebenen Attributen *existenzBedingung* bzw. *fkEindeutigBogenFeld* eingeschränkt werden können.

<sup>19</sup> Falls der Mutterteildatensatz nicht über das Attribut *fkMutterBogen* explizit definiert ist, so gilt implizit der Basisteildatensatz des Moduls als Mutterteildatensatz.

### Inhaltliche Voraussetzung für das Anlegen von Teildatensätzen

Das Attribut `existenzBedingung` ist eine logische Bedingung (Syntax gemäß Abschnitt 9.3.2) für das Anlegen eines Teildatensatzes. Die referenzierten Bogenfelder der Existenzbedingung beziehen sich auf den Mutterteildatensatz.

Die Dokumentationssoftware muss die Existenzbedingung als Trigger für das Anlegen eines abhängigen Teildatensatzes nutzen. Wenn die Existenzbedingung eines potenziellen Kind-Teildatensatzes erfüllt ist, so muss der Kind-Teildatensatz auch angelegt und übermittelt werden. Andererseits gilt: Wenn die entgegennehmende Stelle einen Kind-Teildatensatz erhält, für den die zugehörige Existenzbedingung im Mutterteildatensatz nicht erfüllt ist, so ist das eine relationale Plausibilitätsverletzung.

### Identifizierende Attribute mehrfach vorhandener Teildatensätze

Teildatensätze, die mehr als einmal ausgefüllt werden dürfen (Werte `+` und `*` des Attributs `fkBogenZahl`), sind nicht mehr durch die Vorgangsnummer voneinander unterscheidbar. Diese Teildatensätze benötigen ein zusätzliches identifizierendes Bogenfeld, das im Attribut `fkEindeutigBogenFeld` festgelegt wird.

Beim Anlegen einer Tabelle für die Speicherung eines mehrfach vorhandenen Teildatensatzes muss der Primärschlüssel mindestens die Attribute `Vorgangsnr`<sup>20</sup>, `VersionsNr` und das in `fkEindeutigBogenFeld` definierte Feld umfassen.

Wenn es bei den Teildatensätzen mehr als eine Ebene gibt, muss der Wert des Attributs `fkEindeutigBogenFeld` eines Kind-Bogens eindeutig in Bezug auf den übergeordneten Bogen sein. Hierbei kann sich die Eindeutigkeit des Wertes auf den Elternbogen beschränken, so dass die Kombination beider Werte in Bezug auf den gesamten Vorgang eindeutig ist. Diese Bedingung wird auch erfüllt, wenn das Attribut `fkEindeutigBogenFeld` in Bezug auf den übergeordneten Basisbogen und damit auf den gesamten Vorgang eindeutig ist.

#### 9.2.3 Datenfelder (Bogenfelder)

Jedes auf einem Teildatensatz vorhandene und auszufüllende Feld wird als Datenfeld (Item, Bogenfeld) bezeichnet. Datenfelder sind charakterisiert durch ihren Namen (Bezeichnung) und die Spezifikation des einzutragenden Inhalts.

Die Bezeichnung<sup>21</sup> wird so gewählt, dass sie einem medizinischen Experten unmittelbar verständlich ist. Die Spezifikation des Inhalts umfasst hingegen sowohl eine fachliche (medizinische) als auch datentechnische Typisierung. Dagegen repräsentieren die in der Tabelle `Feld` aufgelisteten

---

<sup>20</sup> Bei den Zusatzfeldern ist zu beachten, dass die Feldnamen beim Export durch die entsprechenden XML-Elemente zu ersetzen sind.

<sup>21</sup> Gegebenenfalls im Kontext der Überschriften (Abschnitt 9.2.4).

Felder inhaltlich gleiche Dokumentationsfelder mehrerer Module (Abschnitt 9.2.1), der datentechnische Typ (`BasisTyp`) charakterisiert das Format des Feldes.

Jedes Datenfeld hat zwingend einen Bezug zu einem Teildatensatz und zu einem technischen Feld. Weitere Eigenschaften sind die Bogenfeldbezeichnung und die fortlaufende Nummer im Teildatensatz. Die Datenfelder sind in der Tabelle `BogenFeld` gespeichert.

Identifizierendes Merkmal eines Datenfelds ist eine Kombination aus `fkBogen` und `fkFeld`. Das bedeutet, dass das Datenbankschema gewährleistet, dass der technische Feldname (`Feld.name`) in einem Teildatensatz maximal einmal vorkommt. Per definitionem muss ein Datenfeldname sogar innerhalb eines Moduls eindeutig sein, d.h. dass eine Abfrage mit dem Primärschlüsselpaar (`idModul`, `idFeld`) genau einen Primärschlüssel `idBogenFeld` liefert.

Tabelle 11: Struktur der Tabelle `BogenFeld`

Feldname	Datentyp	Bemerkung
<code>idBogenFeld</code>	INTEGER	Primärschlüssel
<code>zeileAufBogen</code>	DOUBLE	bestimmt die Reihenfolge von Datenfeldern im Dokumentationsbogen
<code>gliederungAufBogen</code>	TEXT	Gliederungsnummer, die im Dokumentationsbogen angezeigt wird
<code>bezeichnung</code>	TEXT	Beschreibender Text zum Feld auf dem Dokumentationsbogen
<code>ergaenzendeBezeichnung</code>	TEXT	Optionale ergänzende Bezeichnung zu einem Bogenfeld.
<code>elemente</code>	INTEGER	Anzahl der Elemente bei Listenfeldern
<code>fkFeld</code>	INTEGER	Fremdschlüssel zu dem Teildatensatz und zu dem Feld, bilden zusammen die identifizierenden Merkmale
<code>fkBogen</code>	INTEGER	
<code>fkMussKann</code>	TEXT (1)	M oder K, Unterscheidung zwischen Muss- und Kann-Feldern
<code>Min</code>	DOUBLE	Harte Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulspezifisch). Die Definition ist optional.
<code>Max</code>	DOUBLE	Harte Obergrenze des Wertebereichs eines numerischen Datenfeldes (modulspezifisch). Die Definition ist optional.
<code>minWeich</code>	DOUBLE	Weiche Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulspezifisch). Die Definition ist optional.



Feldname	Datentyp	Bemerkung
maxWeich	DOUBLE	Weiche Obergrenze des Wertebereichs eines numerischen Datenfelds (modulspezifisch). Die Definition ist optional.
ahinweis	TEXT (32)	Name des HTML-Ausfüllhinweises ohne Endung .htm (Abschnitt B 9.2.5)
fkICDListe	INTEGER	Zuordnung einer ICD-Liste
fkOPSListe	INTEGER	Zuordnung einer OPS-Liste
berechnungsregel	TEXT	Formel zur Generierung des BogenFeldes, z. B. durch Aufruf einer Syntaxfunktion

### Muss- und Kann-Felder

Jedes Bogenfeld ist als Muss- oder Kann-Feld zu deklarieren:

- Ein Muss-Feld (M) muss innerhalb eines angelegten Teildatensatzes immer ausgefüllt sein (Abschnitt B 9.2.2).<sup>22</sup>
- Kann-Felder (K) sind optionale Felder.
- Abhängige Muss-Felder (K) müssen nur unter bestimmten Bedingungen ausgefüllt werden. Wenn also logische Sachverhalte dem Ausfüllen von Kann-Feldern entgegenstehen, so dürfen sie nicht ausgefüllt werden. Diese Felder unterliegen Feldgruppenregeln und verfügen wie optionale Felder über den Attributwert K.

### Anzahl der Elemente von Listenfeldern

Das Attribut `elemente` ist nur relevant bei von Listenfeldern (vgl. Attribut `istListe` der Tabelle `Feld`) abgeleiteten Bogenfeldern (Bogenfeldlisten). Es gibt die Größe der Bogenfeldliste an. Wenn für eine Bogenfeldliste das Attribut `elemente` leer ist, so ist die Größe per Definition 1.

Wenn ein Listenfeld als Muss-Feld deklariert ist, so ist nur das erste Exportfeld der Liste ein Muss-Feld, die restlichen Elemente sind Kann-Felder. Wenn ein Listenfeld als Kann-Feld deklariert ist, so sind alle weiteren exportierten Elemente ebenfalls Kann-Felder.

### Felder – ein erster Schritt zur Prozess- und Datenintegration

Die Tabelle `Feld` (Tabelle 12) erleichtert dem Softwareanbieter den Abgleich seines Datenmodells mit dem Datenmodell des IQTIG. Gleiche Informationen in der Menge aller Dokumentationsbögen müssen dadurch nicht redundant abgebildet werden.

<sup>22</sup> In jedem Muss-Feld muss für jeden angelegten Teildatensatz einmal eine Angabe erfolgen.

Jedem Feld ist zwingend ein Basistyp zugeordnet. Bei Schlüsselfeldern muss auch ein Schlüssel assoziiert sein. Im Gegensatz zu den (technischen) Basistypen enthalten die Felder die medizinisch-fachliche Information der Datenfelder. Der fachliche Inhalt wird durch den Text im Attribut `bezeichnung`<sup>23</sup> beschrieben.

Identifizierendes Attribut eines Felds ist allein sein technischer Name (Attribut `name`). Dies ist wichtig für die Eindeutigkeit von Feldnamen innerhalb eines Moduls: Felder mit unterschiedlichen Typen oder unterschiedlichen Schlüsseln müssen unterschiedliche Namen haben.

Ein Feld kann als Skalar oder als Liste definiert sein. Diese Eigenschaft wird über das Attribut `istListe` gesteuert. Jedes von einem Listenfeld abgeleitete Bogenfeld ist automatisch eine Liste.<sup>24</sup> Die Anzahl der Elemente des von einem Feld abgeleiteten Bogenfelds wird über das Attribut `elemente` der Tabelle `BogenFeld` gesteuert.

Insbesondere für die Verwendung der richtigen Operatoren in den Plausibilitätsregeln und Feldgruppen ist die Listendefinition eines Felds wichtig.

Grundsätzlich gilt: Die Festlegung, ob ein Bogenfeld ein Skalar oder Listenfeld ist, wird durch die Tabelle `Feld` vorgegeben. Alle von einem Listenfeld abgeleiteten Bogenfelder sind automatisch auch Listenfelder. Die Größe der Liste wird individuell in der Tabelle `BogenFeld` konfiguriert.

Die Tabelle `Feld` bietet über die „Bogensicht“ hinausgehende Informationen.

Tabelle 12: Struktur der Tabelle `Feld`

Feldname	Datentyp	Bemerkung
<code>idFeld</code>	INTEGER	Primärschlüssel
<code>name</code>	TEXT	Technischer Name
<code>bezeichnung</code>	TEXT	(Erlaubte Zeichen: A–Z, 0–9, Ziffer nicht am Anfang) Beschreibender Text auf dem Dokumentationsbogen (Standardwert für gleichnamiges Feld in Tabelle <code>BogenFeld</code> )
<code>laenge</code>	INTEGER	Anzahl der Zeichen in der Feldeingabemaske, enthält beim Typ <code>ZAHL</code> auch das Komma, bei <code>SCHLUESSEL</code> die Trennzeichen
<code>einheit</code>	TEXT (50)	Einheit des Felds (z. B. <code>mm</code> , <code>Stunden</code> )
<code>formatAnweisung</code>	TEXT	Regulärer Ausdruck für die Formatprüfung (z. B. <code>[0–9]{9}</code> )
<code>funktion</code>	TEXT	Formel zur Generierung des Feldinhaltes, z. B. durch Aufruf einer Syntaxfunktion

<sup>23</sup> Das Attribut `bezeichnung` ist ein Standardtext für das gleichnamige Attribut der Tabelle `BogenFeld`. Im Eingabeformular wird die Bezeichnung aus der Tabelle `BogenFeld` angezeigt.

<sup>24</sup> Man beachte die Besonderheiten der Listenfelder beim Datenexport und in der Syntax der Plausibilitätsregeln.

Feldname	Datentyp	Bemerkung
nachKommaLaenge	INTEGER	Anzahl der Nachkommastellen in der Feldeingabemaske (muss kleiner als laenge sein)
min	DOUBLE	Harte Untergrenze des Wertebereichs eines numerischen Datenfelds (modulübergreifend). Die Definition ist optional.
max	DOUBLE	Harte Obergrenze des Wertebereichs eines numerischen Datenfelds (modulübergreifend). Die Definition ist optional.
minWeich	DOUBLE	Weiche Untergrenze des Wertebereichs eines numerischen Datenfelds (modulübergreifend). Die Definition ist optional.
maxWeich	DOUBLE	Weiche Obergrenze des Wertebereichs eines numerischen Datenfelds (modulübergreifend). Die Definition ist optional.
istListe	BOOLEAN	Wenn istListe = WAHR, so sind die vom betreffenden Feld abgeleiteten Bogenfelder Listenfelder.
fkBasisTyp	INTEGER	Fremdschlüssel zur Tabelle Basistypen
fkSchluessel	INTEGER	Fremdschlüssel zur Tabelle Schlüsseltypen
fkKombiFeld	INTEGER	Optionaler Fremdschlüssel auf ein anderes Feld, welches Kombinationsfelder kennzeichnet
strukturParameter	BOOLEAN	Markierung von Strukturparametern, die durch die Software automatisch aus dem KIS/AIS in den Dokumentationsbogen übernommen werden können

### Kombinationsfelder

Für manche Bogenfelder ist zwingend vorgeschrieben, dass sie innerhalb eines Moduls in Kombination mit einem anderen Bogenfeld existieren. Die Definition von Kombinationsfeldern geschieht mithilfe des optionalen Fremdschlüssels `fkKombiFeld` in der Tabelle `Feld`.

### Basistypen

Das Hauptmerkmal eines Basistyps ist der technische Typ eines Eingabefelds (z. B. Zeichenkette, numerischer Typ, Datum usw.). Wichtiges Charakteristikum ist die Beschreibung des Eingabeformats. Die Basistypen sind Voraussetzung für die Beschreibung einer formalen Regelsyntax (Abschnitt 9.3.2).

Das identifizierende Merkmal eines Basistyps ist sein technischer Name (Attribut `name`).

Tabelle 13: Struktur der Tabelle *BasisTyp*

Feldname	Datentyp	Bemerkung
idBasisTyp	INTEGER	Primärschlüssel
Name	TEXT	Technischer Name (muss eindeutig sein)
standardtyp	TEXT	Entsprechender Standarddatentyp
bezeichnung	TEXT	Beschreibender Text
Format	TEXT	Formatdefinition, z. B. TT.MM.JJJJ beim BasisTyp Datum
formatRegExp	TEXT	Regulärer Ausdruck für die Formatprüfung
stdLaenge	INTEGER	Vorschlagsfeld für das gleichnamige Feld in der Tabelle <i>Feld</i> (einschließlich Vorzeichen und Komma)
stdNachKommaLaenge	INTEGER	Vorschlagsfeld für das gleichnamige Feld in der Tabelle <i>Feld</i>

**Hinweis**

- In Zeichenketten (BasisTyp TEXT) sind alle Zeichen des ASCII-Formats mit einem Kode  $\geq 32$  erlaubt. Ausgenommen sind das Semikolon, die doppelten Anführungsstriche und Hochkommata.
- Es gibt zwei Arten von Schlüsseln: numerische und nichtnumerische.
- Das Komma trennt die Nachkommastellen, Vorzeichen + und – sind erlaubt.
- Das Datumstrennzeichen ist der Punkt.

**Schlüssel**

Identifizierendes Merkmal eines Schlüssels (Kodesystem) ist sein technischer Name. Die meisten Schlüsselkodes sind in der Tabelle *SchluesseWert* (Tabelle 15) definiert.

Tabelle 14: Struktur der Tabelle *Schluesse1*

Feldname	Datentyp	Bemerkung
idSchluesse1	INTEGER	Primärschlüssel
name	TEXT	Technischer Name (muss eindeutig sein)
bezeichnung	TEXT	Beschreibender Text
extern	BOOLEAN	Zeigt an, ob der Schlüssel in der Tabelle <i>Schluesse1</i> (= FALSCH) oder in einer externen Tabelle gespeichert (= WAHR) ist.

Feldname	Datentyp	Bemerkung
externVerweis	TEXT	Verweis auf die Quelle des externen Schlüssels
zahl	BOOLEAN	Wenn WAHR, sind die Werte im Attribut <code>code</code> der zugehörigen Schlüsselwerte als ganze Zahl kodiert, ansonsten als Zeichenkette.
sortierNrVerwendet	BOOLEAN	Flag, das anzeigt, ob für die Reihenfolge das Attribut <code>sortierNr</code> der Tabelle <code>SchlüsselWert</code> herangezogen wird.
fkMutterSchluessel	INTEGER	Referenz auf einen übergeordneten Schlüssel.

Schlüsselcodes können auf zwei Arten interpretiert werden: Wenn das Attribut `zahl` gesetzt ist, so werden die Codes als ganze Zahl gedeutet, ansonsten werden sie als Zeichenketten interpretiert. In der Syntax der Plausibilitätsregeln werden die letztgenannten Codes in einfache Hochkommata gesetzt (Abschnitt 9.3.2).

---

#### Beispiel:

Attribut `zahl` bei Schlüsselfeldern

- Felder des Basistyps `NUMSCHLUESSEL` haben das Attribut `zahl = WAHR`.
- Felder des Basistyps `SCHLUESSEL` haben das Attribut `zahl = FALSCH`. Es handelt sich um alphanumerische Schlüssel, die Buchstaben, Ziffern oder Sonderzeichen verwenden (z. B. `ypN0`). Hierbei kann es sich auch um Werte handeln, die lediglich Ziffern verwenden, aber mit einer führenden Null beginnen (z. B. `01`).

---

### Externe Schlüsselkataloge

Externe Schlüsselkataloge sind über das Attribut `extern` deklariert. Hinweise zu den Bezugsquellen sind in der Spalte `externVerweis` zu finden. Diese externen Schlüsselkataloge werden nicht vom IQTIG bereitgestellt und somit auch nicht verantwortet.



#### Achtung

Der Softwareanbieter hat dafür Sorge zu tragen, dass die jeweils aktuellen externen Schlüsselkataloge in der Software verwendet werden.

Die datenentgegennehmenden Stellen müssen ebenfalls die aktuellen Schlüsselkataloge verwenden und fehlerhafte Datensätze abweisen.

---

Hinweise zu den Bezugsquellen sind in der Spalte `externVerweis` zu finden. Ein Verweis auf eine Bezugsquelle kann unabhängig vom Attribut `extern` angegeben werden.

Die Schlüsselcodes sind in der Tabelle `SchlüsselWert` enthalten. Spätere Schlüsseländerungen bzw. -fortschreibungen werden vom IQTIG zeitnah übernommen.

## Schlüsselwerte

Tabelle 15 gibt einen Überblick über die Datenbanktabelle `SchlüsselWert`, in der die Codes und Bezeichnungen der Schlüssel hinterlegt sind. Identifizierendes Merkmal ist hier eine Kombination der Spalten `fkSchlüssel` und `code`. Das bedeutet, dass jeder Schlüsselcode innerhalb eines Schlüssels nur einmal vorkommen darf.

Tabelle 15: Struktur der Tabelle `SchlüsselWert`

Feldname	Datentyp	Bemerkung
<code>idSchlüsselWert</code>	INTEGER	Primärschlüssel
<code>fkSchlüssel</code>	INTEGER	Fremdschlüssel zur Tabelle <code>Schlüssel</code>
<code>code</code>	TEXT (50)	Schlüsselcode (entweder numerisch oder alphanumerisch kodiert)
<code>bezeichnung</code>	TEXT	Textliche Definition des Schlüsselwertes
<code>sortierNr</code>	INTEGER	Optionale Angabe zur Reihenfolge der Schlüsselwerte: Wenn belegt, so ist diese Reihenfolge bei der Anzeige in der Erfassungssoftware einzuhalten.

Das Attribut `code` der Tabelle `SchlüsselWert` ist ein Textfeld, das in Abhängigkeit vom Wert des Attributes `zahl` im zugeordneten Schlüssel entweder numerisch oder nichtnumerisch interpretiert wird. Wenn in einer Plausibilitätsregel (Abschnitt 9.3.2) Felder mit numerischen Schlüsseln (Basistyp `NUMSCHLUESSEL`) vorkommen, so werden bei der Evaluierung der Regel die Schlüsselcodes wie ganze Zahlen behandelt.

## Sortierung der Codes

- Für die Codes (Attribut `SchlüsselWert.code`) eines Schlüssels ist eine Sortierung definiert. Die Art der Sortierung wird über die Attribute `zahl` und `sortierNrVerwendet` der Tabelle `Schlüssel` festgelegt.
- Numerische Sortierung: Wenn `sortierNrVerwendet = FALSCH` und `zahl = WAHR`, so sind die Codes nach der Spalte `code` der Tabelle `Schlüssel` numerisch zu sortieren.
- Alphanumerische Sortierung: Wenn `sortierNrVerwendet = FALSCH` und `zahl = FALSCH`, so sind die Codes nach der Spalte `code` der Tabelle `Schlüssel` alphanumerisch zu sortieren.
- Spezielle Sortierung: Wenn `sortierNrVerwendet = WAHR`, so sind die Codes nach den Werten in der Spalte `sortierNr` der Tabelle `Schlüssel` numerisch zu sortieren. Suchfunktion bei Schlüsseln mit einer großen Anzahl von Codes

**Hinweis**

Das Feld `sortierNr` ist dann befüllt, wenn im Feld `Code` ein alphanumerischer Wert enthalten ist.

Bei Schlüsseln mit einer großen Anzahl von Codes soll eine anwenderfreundliche Möglichkeit zur Auswahl der passenden Codes bereitgestellt werden. Die Umsetzung als Auswahlliste (z. B. Combobox) führt zu erhöhtem Dokumentationsaufwand, falls der Anwender über Pfeiltasten oder Schieberegler zum passenden Code navigieren muss. Ergänzend soll daher eine Suchfunktion realisiert werden, die eine Suche über die Attribute `SchlüsselWert.code` oder `SchlüsselWert.bezeichnung` ermöglicht.

### 9.2.4 Überschriften vom Dokumentationsbogen

Die Überschriften der Dokumentationsbögen in der Spezifikation sind in der Tabelle `Abschnitt` zu finden.

Tabelle 16: Struktur der Tabelle `Abschnitt`

Feldname	Datentyp	Bemerkung
<code>idAbschnitt</code>	INTEGER	Primärschlüssel
<code>bezeichnung</code>	TEXT	Text der Überschrift
<code>Ebene</code>	INTEGER	Zeigt die Hierarchie der Überschriften an
<code>fkStartBogenFeld</code>	INTEGER	Fremdschlüssel auf das erste zur Überschrift gehörende Bogenfeld
<code>fkEndeBogenFeld</code>	INTEGER	Fremdschlüssel auf das letzte zur Überschrift gehörende Bogenfeld

Zu jeder Überschrift ist angegeben, bei welchem Bogenfeld sie beginnt und bei welchem Bogenfeld sie endet. Über das Attribut `ebene` lassen sich auch Teilüberschriften realisieren. Ein Bogenfeld kann somit mehreren Überschriften zugeordnet sein.

**Achtung**

Die in der Spezifikationsdatenbank hinterlegten Überschriften sind in die Eingabemasken der Software zu integrieren. Viele Datenfelder sind für den Anwender erst im Kontext der Überschriften verständlich.

### 9.2.5 Ausfüllhinweise

Die Ausfüllhinweise zu den Datenfeldern sind in einem separaten ZIP-Archiv enthalten, das nach dem Benennungsschema für Spezifikationskomponenten bezeichnet wird. Jeder Ausfüllhinweis ist ein HTML-Dokument.

In der Spalte `ahinweis` der Tabelle `BogenFeld` ist festgelegt, welcher HTML-Ausfüllhinweis mit einem Datenfeld verknüpft ist:

`<aHinweis>.htm` = Name der HTML-Datei

Wenn der Eintrag in `ahinweis` leer ist, so existiert für das betreffende Bogenfeld kein Ausfüllhinweis. Das Attribut `fkAhinweisTyp` lässt die Differenzierung drei verschiedener Arten von Ausfüllhinweisen zu:

Tabelle 17: Arten von Hinweistypen

<code>fkAhinweisTyp</code>	Beschreibung	Beispiele (aus PPP)
Feldbezogen	Der Ausfüllhinweis bezieht sich auf den entsprechenden Datensatz in der Tabelle <code>Feld</code> . Der Ausfüllhinweis ist modulunabhängig.	<code>VKSISTMIN.htm</code> Der Ausfüllhinweis bezieht sich auf verschiedene Module.
Modulspezifisch	Soll sich ein Ausfüllhinweis nur auf ein bestimmtes Modul beziehen, kann der Attributwert modulspezifisch ausgewählt werden.	<code>IDSTATION\$PPP.htm</code> Der Ausfüllhinweis bezieht sich nur auf das Modul PPP.
Speziell	Soll es für verschiedene Datenfelder der Tabelle <code>Feld</code> einen gemeinsamen Ausfüllhinweis geben, kann dieser als speziell deklariert werden. Der Attributwert <code>ahinweis</code> definiert den Namen des Ausfüllhinweises.	<code>BEHBEREICH.htm</code> Die Felder <code>BEHBEREICHEP</code> , <code>BEHBEREICHPS</code> und <code>BEHBEREICHKJP</code> haben denselben Ausfüllhinweis.

Die Zuordnung von Bogenfeldern und Ausfüllhinweisen ist auch in der Abfrage `Ausfüllhinweise` dargestellt. Sie zeigt Modul/Teildatensatz, Zeile, Feldname, Bezeichnung und den HTML-Dateinamen des Ausfüllhinweises zu dem Bogenfeld. Im Gegensatz zur Tabelle `Bogenfeld` ist hier die Endung `.htm` mit angegeben.

### 9.3 Plausibilitätsprüfungen

Es wird zwischen drei Arten von Plausibilitätsprüfungen unterschieden, die in Tabelle `RegelTyp` definiert sind (siehe nachstehende Tabelle 18):

- Harte Prüfungen
- Weiche Prüfungen in der QS-Dokumentationssoftware

Tabelle 18: Tabelle `RegelTyp`

<code>idRegelTyp</code>	bezeichnung
H	Hart
W	Weich



Weiterhin wird zwischen sogenannten **Einzelregeln** (Abschnitt 9.3.4) und **Feldgruppenregeln** (Abschnitt 9.3.8) unterschieden.

### 9.3.1 Die Regeltabelle

Die Bedingungen für nicht plausible Angaben sind in der Tabelle `Regeln` abgelegt. Die hier beschriebenen Prüfungen sind in der Spezifikationsdatenbank für QS-Dokumentation hinterlegt. Die Syntax ist in Abschnitt 9.3.4 beschrieben. Die Bedingungen sind möglichst kurzgefasst (Vermeidung von durch `ODER` verknüpften Teilbedingungen). Jede Bedingung kommt nur einmal innerhalb eines Moduls vor.

Tabelle 19: Struktur der Tabelle `Regeln`

Feldname	Datentyp	Bemerkung
<code>idRegeln</code>	INTEGER	Primärschlüssel
<code>fkModul</code>	INTEGER	Fremdschlüssel zur Tabelle <code>Modul</code>
<code>bedingung</code>	MEMO	Entsprechend der Syntax definierte Regeln
<code>meldung</code>	MEMO	Fehlermeldung: Diese Texte sind bei Regeln mit Bezug zu Feldgruppen automatisch generiert.
<code>alternativMeldung</code>	MEMO	Alternative Fehlermeldung: Wenn hier ein Text vorhanden ist, so ist dieser anstelle des Textes in der Spalte <code>meldung</code> zu verwenden.
<code>fkMehrfachRegel</code>	INTEGER	Fremdschlüssel zur Tabelle mit mehrfach vorkommenden Regeln, die mithilfe von Ersatzbedingungen nach dem Export gültig sind.
<code>fkFeldGruppe</code>	INTEGER	Optionaler Fremdschlüssel zur Tabelle <code>FeldGruppe</code> : Indikator dafür, ob eine Regel aus einer Feldgruppe generiert wurde.
<code>fkRegelTyp</code>	TEXT (1)	Fremdschlüssel zur Tabelle <code>RegelTyp</code> : Die Regeltypen sind die in Abschnitt 9.3 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D
<code>gueltigNachExport</code>	BOOLEAN	Regeln, die den Wert <code>FALSCH</code> haben, können von Datenannahmestellen nicht evaluiert werden. Stattdessen werden die referenzierten Ersatzbedingungen der Tabelle <code>MehrfachRegel</code> evaluiert (falls definiert).

#### Regelfelder (Bogenfelder einer Regel)

Die Tabelle `RegelFelder` (Tabelle 20) ist eine Verknüpfungstabelle zwischen den Tabellen `Regeln` und `BogenFeld`. Durch gezielte Abfragen erhält man unter Verwendung dieser Tabelle einen Überblick über Folgendes:

- Bogenfelder, die in einer Regel verwendet werden.
- Regeln, die sich auf ein Bogenfeld beziehen.

Tabelle 20: Struktur der Tabelle *RegelFelder*

Feldname	Datentyp	Bemerkung
fkBogenFeld	INTEGER	Fremdschlüssel zu den Tabellen <i>Feld</i> und <i>Regeln</i> , bilden zusammen den Primärschlüssel
fkRegeln	INTEGER	

### Mehrfachregeln (Ersatzregeln zur Prüfung nach dem Export)

Wenn in einer Regel von der Pseudonymisierung betroffene Datenfelder benutzt werden, so kann diese von Datenannahmestellen nicht evaluiert werden. Stattdessen wird für solche Regeln in der Tabelle *MehrfachRegel* eine Ersatzbedingung definiert, deren Referenzierung in der Tabelle *Regeln* definiert ist. Die Ersatzbedingung ist von den Datenannahmestellen zu evaluieren.

Tabelle 21: Struktur der Tabelle *MehrfachRegel*

Feldname	Datentyp	Bemerkung
idMehrfachRegel	INTEGER	Primärschlüssel
bedingung	MEMO	Entsprechend der Syntax definierte Regeln
meldung	MEMO	Kontextunabhängige Fehlermeldung
ersatzBedingung	MEMO	Ersatzregel für den pseudonymisierten Falken1980Datensatz
fkRegelTyp	TEXT (1)	Fremdschlüssel zur Tabelle <i>RegelTyp</i> : Die Regeltypen sind die in Abschnitt 9.3 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D

### Weitere Regeln

Weitere feldübergreifende Regeln sind die in Abschnitt 9.2.2 beschriebenen Existenzbedingungen für das Anlegen von abhängigen Teildatensätzen (Attribut *existenzBedingung* in Tabelle *Bogen*).

### 9.3.2 Regelsyntax

Bedingungen sind in den Tabellen *Regeln*, *MehrfachRegel* und *Bogen* definiert. Die den Bedingungen zu Grunde liegende Regelsyntax wird in diesem Abschnitt beschrieben. Jede Regel ist ein logischer Ausdruck, dessen Ergebnis WAHR oder FALSCH lautet. Jede Regel bezieht sich auf einen eingegebenen Datensatz eines Moduls, dessen Daten in Variablen gespeichert sind.

Die Regelsyntax lehnt sich an die logischen Ausdrücke in bekannten Programmiersprachen an. Jedoch haben die Operatoren deutsche Namen, z. B. UND statt AND oder ODER statt OR. Die Regelsyntax ist als Pseudocode zu verstehen.

## Typen

Die möglichen Typen der Datenfelder sind in Tabelle 22 aufgelistet.

Tabelle 22: Basistypen der Datenfelder in den Plausibilitätsregeln

Basistyp	Bezeichnung	Beispiele (Literele)
BOOL	Boolesche Variable	WAHR, FALSCH
TEXT	Zeichenkette (String)	"Spezifikation"
GANZEZAHL <sup>25</sup>	... -2, -1, 0, 1, 2, 3, ...	1
ZAHL	Zahl (mit oder ohne Nachkommastellen)	25,4 oder -100,8
DATUM	Zehnstelliges Datum	'01.01.2012'
MONDATUM	Monatsdatum	'04.2012'
QUARTDATUM	Quartalsdatum	'3/2012'
JAHRDATUM	Jahresdatum	2012
NUMSCHLUESSEL	Numerisch kodierter Schlüssel (wie GANZEZAHL)	1
SCHLUESSEL	Alphanumerischer Schlüssel	'19.1', '07'
UHRZEIT	Uhrzeit	'10:15'

In der Spezifikation für die QS-Dokumentation wird zwischen NUMSCHLUESSEL und SCHLUESSEL unterschieden:

- Schlüsselwerte verfügen über den Datentyp NUMSCHLUESSEL, wenn es sich bei den Codes um ganze Zahlen handelt. Da dies ein numerischer Schlüssel ist, darf er nicht in Hochkommata gesetzt werden.
- Schlüsselwerte, die alphanumerische Codes beinhalten, haben den Basistyp SCHLUESSEL. Die OPS-Schlüssel (z. B. '5-282.0') und die ICD-10-GM-Schlüssel verfügen über diesen Datentyp, der in Hochkommata geschrieben wird.



### Achtung

Datumsangaben (Datum, Monats-, Quartalsdatum) müssen in Hochkommata gesetzt werden. Eine Ausnahme ist das Jahresdatum (JAHRDATUM), da es sich hierbei um eine ganze Zahl handelt.

<sup>25</sup> Beim Typ GANZEZAHL sind auch negative ganze Zahlen erlaubt.

## Felder

Feldnamen bestehen aus maximal 32 Zeichen und dürfen nur die Buchstaben A bis Z (Großbuchstaben) und die Ziffern 0 bis 9 enthalten. Ein Feldname muss immer mit einem Buchstaben beginnen. Umlaute und Sonderzeichen sind in Feldnamen nicht erlaubt. Ein Feldname darf kein reserviertes Wort sein (z. B. `LEER`).



### Achtung

In einer Regel dürfen nur die Feldnamen der im betreffenden Modul definierten Bogenfelder<sup>26</sup> enthalten sein. Bei der Evaluierung von Regeln werden die aktuellen Werte der referenzierten Bogenfelder eingesetzt. Kann-Bogenfelder können auch unausgefüllt sein, also den Wert `LEER` haben.

---

## Listenfelder

Ein Bogenfeld wird dann als Liste interpretiert, wenn im referenzierten Feld (Tabelle `Feld`) der Wert des Attributs `Feld.istListe = WAHR` ist. Andernfalls ist das Bogenfeld ein Skalar. Bei der Formulierung von Regeln ist darauf zu achten, dass Listenfelder nicht bei jedem Operator als Operand fungieren können. Listenfelder dürfen z. B. nicht voneinander subtrahiert werden.

## Literale

Alphanumerische Literale (z. B. `SCHLUESSEL`) werden von einfachen Hochkommata eingeschlossen, während Zeichenketten vom Datentyp `TEXT` in Anführungszeichen gesetzt werden müssen.<sup>27</sup>

Dies gilt nicht für numerische Literale (`GANZEZAHL`, `ZAHL`, `NUMSCHLUESSEL` und `JAHRDATUM`) und Literale des Datentyps `BOOL` (Wahrheitswerte).

## Listen von Literalen

Literale können sowohl als Skalare als auch als Listen angesprochen werden. Der Separator einer Liste von Literalen ist das Semikolon. Um zu prüfen, ob alle Listenfelder ausgefüllt sind, wird die Liste über den Wert `LEER` angesprochen.

---

### Beispiele für Listen von Literalen:

- Liste von Literalen vom Typ `GANZEZAHL` oder `NUMSCHLUESSEL`:  
`(1;2;3)`
  - Liste von Literalen vom Typ `SCHLUESSEL` (alphanumerisch):  
`('5-740.0'; '5-740.1'; '5-740.y')`
- 

---

<sup>26</sup> Bei den Ersatzregeln in Tabelle `MehrfachRegel` sind stattdessen die Exportfelder des Moduls erlaubt.

<sup>27</sup> Beim Export entfallen die begrenzenden Zeichen.

Längere Listen von Prozedurcodes (OPS) oder Diagnosecodes (ICD-GM-10) werden als Variable angesprochen, deren Namen einem festen Namensschema gehorchen. Diese Listen werden in separaten Tabellen definiert, die den Variablennamen<sup>28</sup> und die darin enthaltenen Prozedur- und Diagnosecodes beinhalten (Abschnitt 9.4). Außerdem gibt es Teildatensatz-Listenfelder, die im Abschnitt 9.3.6 beschrieben werden.

## Operatoren

Tabelle 23 gibt einen Überblick über die in der Syntax zulässigen Operatoren. Der aktuelle Überblick über alle zulässigen Operationen (inkl. Operanden) ist in Tabelle `SyntaxOperator` in der QSDOK-Datenbank zu finden.

In Tabelle 23 hat jeder einzelne Operator eine Präzedenzstufe (höchste Präzedenzstufe ist 0). Operatoren, die die gleiche Stufe haben, werden nach den Regeln der Assoziativität aufgelöst.

Tabelle 23: Präzedenz und Assoziativität der Operatoren

Präzedenz	Assoziativität	Operator	Erläuterung
0	links	IN	Operator zum Vergleich einer Variablen mit einer Liste (z. B. ein Datenfeld mit Schlüsselwerten). Die Variable und die Feldelemente müssen gleichen Typs sein.
	links	NICHTIN	
	links	EINSIN	Operator zum Vergleich einer Liste mit einer anderen Liste oder einem Listenelement (z. B. ein Listenfeld mit einem Schlüsselwert oder ein Listenfeld mit einer OPS-Liste). Die Listenelemente müssen gleichen Typs sein.
	links	JEDESIN	
	links	EINSNICHTIN	
	links	KEINSIN	
1	links	*	Operator für die Multiplikation „mal“
	links	/	Operator für die Division „geteilt“
2	links	+	Operator für die Addition „plus“
	links	-	Operator für die Subtraktion „minus“
3	links	<	Vergleichsoperator „kleiner“
	links	>	Vergleichsoperator „größer“
	links	<=	Vergleichsoperator „kleiner gleich“
	links	>=	Vergleichsoperator „größer gleich“
4	links	=	Vergleichsoperator „gleich“
	links	<>	Vergleichsoperator „ungleich“
5	rechts	NICHT	Logischer Operator: „NICHT“

<sup>28</sup> Der Variablenname ist synonym mit dem Listennamen (z. B. `OPSListe.name`) der Prozedur- bzw. Diagnosenliste.

Präzedenz	Assoziativität	Operator	Erläuterung
6	links	UND	Logischer Operator: „UND“
7	links	ODER	Logischer Operator: „ODER“

### Prüfung auf LEER mit Vergleichsoperatoren

Die Prüfung auf `LEER` von in Regeln verwendeten Kann-Feldern, welche an anderer Stelle in der Regel mit einem anderen Operator als `<>` oder `=` geprüft werden sollen, findet auf der linken Seite einer `ODER`-Verknüpfung statt. Hintergrund dieser Syntaxregel ist, dass die Vermeidung von Laufzeitfehlern bei der Evaluation ermöglicht werden soll.

---

#### Beispiel:

`FELD = LEER ODER FELD OPERATOR OPERAND`

Beispielsweise kann bei leeren Feldwerten und der vorgegebenen Linksassoziativität des `ODER`-Operators die Evaluation bei leerem Feldwert vor der Evaluation des rechtsstehenden Ausdrucks mit der Rückgabe von `WAHR` abgebrochen werden. Ein Laufzeitfehler, der sich z. B. bei einem Vergleich von `LEER < WERT` ergeben würde, kann so nicht entstehen.

---

### Operatoren zum Vergleich einer Variablen mit einer Liste

Folgende Operatoren erfordern entweder nur rechts oder links und rechts Listenfelder:

- nur rechts: `IN`, `NICHTIN`
- links und rechts: `EINSIN`, `KEINSIN`, `JEDESIN`, `EINSNICHTIN`

Operatoren mit beidseitigen Listenfeldern als Operanden:

- `EINSIN`: Wenn mindestens ein Element aus der linken Liste in der rechten Liste enthalten ist, so ist der Ausdruck wahr (nichtleere Schnittmenge).
  - `KEINSIN`: Wenn kein Element der linken Liste in der rechten Liste enthalten ist, so ist der Ausdruck wahr (leere Schnittmenge). Dieser Operator ist redundant, da er auch durch Negation des `EINSIN`-Operators abgedeckt ist.
  - `JEDESIN`: Der Ausdruck ist dann wahr, wenn jedes Element der linken Liste in der rechten Liste enthalten ist (Teilmenge).
  - `EINSNICHTIN`: Der Ausdruck ist dann wahr, wenn mindestens ein Element der linken Liste nicht in der rechten Liste enthalten ist (nichtleere Differenz).
- 

#### Beispiel:

- Die Operation `GANZEZAHL := DATUM1 - DATUM2` liefert als Ergebnis die Differenz zwischen zwei Kalenderdaten in Tagen.
- Die Operation `ZAHL := UHRZEIT1 - UHRZEIT2` liefert als Ergebnis die Differenz zwischen zwei Uhrzeiten in Minuten.

---

**Beispiel:**

Folgende Regel prüft, ob kein Element des Listenfeldes OPSCHLUESSEL (4 Elemente) einen bestimmten Kode besitzt:

```
OPSCHLUESSEL KEINSIN ('5-983')
```

Wenn z. B.

```
OPSCHLUESSEL := ('5-661.3y';LEER;LEER;LEER),
```

so ist die Regel erfüllt. Gleichwertig ist die Regel:

```
NICHT OPSCHLUESSEL EINSIN ('5-983')
```

Eine Besonderheit bei Listenoperationen ist die Prüfung, ob alle Elemente einer Liste ausgefüllt sind:

---

**Beispiel:**

```
NICHT OPSCHLUESSEL JEDESIN (LEER)
```

Diese Bedingung erfordert, dass zumindest ein Listenelement ausgefüllt ist. Beispielsweise erfüllt

```
OPSCHLUESSEL := ('5-661.3y';LEER;LEER;LEER)
```

die Bedingung. Gleichwertig ist die Regel:

```
OPSCHLUESSEL EINSNICHTIN (LEER)
```

---

Folgende Operatoren sind komplementär:

IN und NICHTIN

EINSIN und KEINSIN

JEDESIN und EINSNICHTIN

Folgende Ausdrücke sind gleich:

A EINSNICHTIN B

NICHT (A JEDESIN B)

---

**Plausibilitätsprüfungen mit OPS- und ICD-Listen**

Die OPS- und ICD-Listen enthalten ausschließlich Normkodes. Die vom Leistungserbringer dokumentierten Codes enthalten ggf. auch Zusatzkennzeichen (Bsp.: Seitenlokalisierung). Bei der Evaluation der Regeln werden die dokumentierten Zusatzkennzeichen ignoriert (Abschnitt B 9.4).

**Hinweis**

---

Wird das Zusatzkennzeichen direkt in der Plausibilitätsregel abgefragt, wird dieses bei der Evaluation der entsprechenden Regel nicht ignoriert. Hierbei kann es sich beispielsweise um die Prüfung der Diagnosesicherheit mithilfe der Funktion `format` handeln.

---

### 9.3.3 Funktionen

Eine Funktion ist gekennzeichnet durch ihren Namen, an den sich unmittelbar (ohne Leerzeichen) ein Listenausdruck anschließt. Funktionen ohne Übergabeparameter werden ähnlich wie in Code Java durch ein Klammerpaar abgeschlossen. Funktionen können nicht nur in Regeln, sondern auch zur Berechnung von Exportfeldern genutzt werden (Abschnitt 9.5.2).

Der aktuelle Stand der in der Syntax verwendeten Funktionen ist in der Tabelle `SyntaxFunktion` der Spezifikation zu finden.

In den nachfolgenden Beispielen gilt folgende Notation für Funktionen:

```
<BASISTYP> <FUNKTIONSNAME> ([<BASISTYP> <VARNAME>{ ;
<BASISTYP> <VARNAME>}])
```

mit

- { }                   Wiederholung
- [ ]                   Option
- <BASISTYP>          Basistyp der Variablen
- <VARNAME>          Name der Variablen

---

#### Beispiele:

```
DATUM aktuellesDatum()
```

Funktion ohne Übergabeparameter und mit Ergebnistyp `DATUM`

```
DATUM Minimum(DATUM DATUMLISTE)
```

Funktion mit Ergebnis vom Typ `DATUM`, die das Minimum einer Liste von Datumsangaben (`DATUMLISTE`) liefert.

```
JAHRDATUM jahreswert(DATUM EINDATUM)
```

Funktion mit Ergebnis vom Typ `JAHRDATUM`

---

Es kommen auch verschachtelte Funktionsaufrufe (z. B. `funktionA(funktionB())`) oder arithmetische Ausdrücke als Funktionsargumente (z. B. `funktion(x+y)`). Häufig wird nur die Signatur von Funktionen bereitgestellt.

#### Hinweise für die Implementierung von Funktionen

Als Hilfestellung für die Ausprogrammierung wird bei manchen Funktionen ein Pseudocode bereitgestellt. Der Pseudocode ergänzt die Syntax der Plausibilitätsregeln um folgende Sprachelemente:<sup>29</sup>

---

<sup>29</sup> Der Pseudocode erhebt nicht den Anspruch auf formale Korrektheit.



- Befehlszeilen werden mit Semikolon abgeschlossen ;
- Wertzuweisungen mit dem Operator `:= A := B + C;`
- Auswahlanweisungen

```
if (<Bedingung>) {
    ...
}
else {
    ...
}
```

Hinter `<Bedingung>` verbirgt sich ein logischer Ausdruck, der der Syntax der Plausibilitätsregeln gehorcht.

- Blöcke werden durch geschweifte Klammern definiert.

```
{
...
}
```

- Innerhalb einer Funktion sind die Argumentvariablen verfügbar.

Eine Variable, die den gleichen Namen wie die Funktion hat, muss am Ende mit `return` zurückgegeben werden.

#### Hinweise zur Funktion `format (Feld, pattern)`

Die Funktion prüft, ob der erste Parameter (`Feld`) dem regulären Ausdruck (`pattern`) entspricht. Gibt es eine Übereinstimmung, gibt die Funktion ein `WAHR` zurück. Die konkrete Implementierung dieser Funktion ist von der eingesetzten Programmiersprache abhängig.

JAVA	C#	C++
<code>feld.matches (pattern)</code>	<code>Regex.IsMatch (feld,pattern)</code>	<code>Regex::IsMatch (feld,pattern)</code>

#### Hinweise zur Funktion `verkettenmt`

Die Funktion verkettet (zwei oder mehrere) Zeichenfolgen zu einer Zeichenfolge. Dabei wird die als erster Parameter übergebene `TRENNZEICHENFOLGE` zwischen alle nicht leeren Elemente der `TEXTLISTE` eingesetzt, d.h. sollte ein Element der `TEXTLISTE` leer sein, findet für dieses keine Verkettung mit der `TRENNZEICHENFOLGE` statt.



#### Achtung

Ist lediglich eine übergebene Zeichenfolge nicht leer, wird diese ohne Verkettung zurückgeliefert.

---

### 9.3.4 Syntaxvariablen

Syntaxvariablen in der QS-Dokumentation dienen der technischen Darstellung der automatischen Generierung von Angaben aus dem Eingangsdatensatz. Diese sind in Tabelle `SyntaxVariable` hinterlegt. Jedes dieser Felder besitzt somit einen Basistyp.

Die in den Bedingungen erlaubten Variablen sind in der Tabelle `SyntaxVariable` definiert.

Die Variablennamen (Attribut `SyntaxVariable.name`) bestehen aus maximal 32 Zeichen. Sie dürfen nur die Buchstaben A bis Z (Großbuchstaben) und die Ziffern 0 bis 9 enthalten. Ein Feldname muss immer mit einem Buchstaben beginnen. Umlaute und Sonderzeichen sind in Feldnamen nicht erlaubt. Ein Feldname darf auch kein reserviertes Wort sein (z. B. Namen von Operatoren wie `EINSIN`).

### 9.3.5 Einzelregeln

Sogenannte Einzelregeln können sich als feldbezogene Prüfungen auf ein einziges Datenfeld oder als feldübergreifende Prüfungen auf mehrere Datenfelder beziehen. Einzelregeln sind von den in Abschnitt B 9.3.8 beschriebenen Feldgruppen zu unterscheiden.

**Feldbezogene Prüfungen** – beispielsweise Wertebereichsüberprüfungen – sind in der formalen Regelsyntax in Tabelle **Regeln** formuliert.

Unter feldbezogenen Prüfungen sind aber auch die beschriebenen Prüfungen des Formates, der Feldlänge, der Wertebereiche, Prüfungen von Schlüsselcodes und von Muss-Feldern zu verstehen. Für diese Prüfungen gibt es keine formale Regelsyntax in Tabelle `Regeln`.

#### **Feldübergreifende Regeln**

- haben eine eigene Syntax,
- haben geringe Komplexität,
- haben einfache, dem Anwender verständliche Fehlertexte,
- enthalten alle Teilregeln der Feldgruppen,
- haben gewöhnlich den Bezug zu zwei oder mehreren Feldern,
- können zum Teil direkt nach der Benutzereingabe in ein Feld geprüft werden,
- enthalten Bedingungen für unplausible Angaben<sup>30</sup>.

Feldübergreifende Regeln können auch teildatensatzübergreifende Regeln sein, wenn die Datenfelder der Regel aus mehreren Teildatensätzen eines Moduls stammen (Abschnitt 9.3.6).

### 9.3.6 Teildatensatzübergreifende Regeln

Eine Regel ist teildatensatzübergreifend, wenn die Datenfelder der Regel aus mehreren Teildatensätzen eines Moduls stammen.

---

<sup>30</sup> Eine Plausibilitätsregel müsste eigentlich „Unplausibilitätsregel“ heißen, weil sie unplausible Zustände beschreibt, die zu Fehlermeldungen führen.

Es gibt zwei Arten von teildatensatzübergreifenden Regeln:

- Die Felder sind in verschiedenen Teildatensätzen eines Moduls definiert.
- Ein Feld der Regel ist in einem wiederholbaren Teildatensatz definiert und die Regel bezieht sich auf alle Werte des Datenfeldes innerhalb eines Datensatzes (= Summe aller Teildatensätze eines Vorgangs).

### 9.3.7 Regeln mit Teildatensatz-Listenfeldern

Zu jedem skalaren Datenfeld eines wiederholbaren Teildatensatzes existiert ein Teildatensatz-Listenfeld (kurz TDS-Listenfeld), das über das @-Zeichen vor dem Feldnamen angesprochen wird. Das TDS-Listenfeld enthält sämtliche Werte des betreffenden Datenfeldes, die innerhalb der QS-Dokumentation eines Vorgangs existieren Feldgruppenregeln.

### 9.3.8 Feldgruppenregeln

Logische Abhängigkeiten von Bogenfeldern werden über Feldgruppen dargestellt. Die Plausibilitätsregeln, die einen Bezug zu einer Feldgruppe aufweisen (Tabelle `Regeln`), werden anhand der Feldgruppendefinition (Tabelle `FeldgruppeFelder`) automatisch generiert. Die Menge der abgeleiteten Einzelregeln wird in diesem Abschnitt erläutert.

Die möglichen Antworten<sup>31</sup> eines jeden Datenfeldes werden in zwei Gruppen aufgeteilt. Die erste Gruppe ist die Menge der positiven, die zweite Gruppe die Menge der negativen Antworten.<sup>32</sup>

Typische positive Antworten sind beispielsweise:

```
Feld <> LEER oder Feld IN (2;3)
```

Die komplementären negativen Antworten würden entsprechend wie folgt lauten:

```
Feld = LEER oder Feld NICHTIN (2;3)
```

Eine Feldgruppe kann ein Filterfeld haben. Wenn die Antwort dieses Filterfeldes negativ ausfällt (bspw. Bedingung: `Feld = 3`; Antwort: `Feld <> 3`), so darf keines der abhängigen Felder positiv beantwortet werden. Die folgende Tabelle gibt einen Überblick über die Typen von Feldgruppen. Der aktuelle Stand findet sich in der Tabelle `FeldGruppenTyp` der Spezifikation.

Tabelle 24: Typen von *Feldgruppen*

Name	Bemerkung
<b>mit Filterfeld</b>	
EF_FILTER	Einfachauswahl, genau ein abhängiges Feld muss positiv beantwortet sein

<sup>31</sup> Die Antworten eines Datenfeldes umfassen hier neben möglichen Werten (z. B. Schlüsselwerten) oder Wertemengen auch die Kategorie „nicht ausgefüllt“ (LEER).

<sup>32</sup> Die negativen Antworten sind abhängig von der definierten Bedingung eines Feldes in der entsprechenden Feldgruppe.

EF_OPTIONAL_FILTER	Einfachauswahl, genau ein abhängiges Feld kann positiv beantwortet sein
MF_OPTIONAL_FILTER	Mehrfachauswahl, alle abhängigen Felder können positiv beantwortet sein
MF_MINDESTENS1_FILTER	Mehrfachauswahl, mindestens ein abhängiges Feld muss positiv beantwortet sein
MF_ALLES_FILTER	Mehrfachauswahl, alle abhängigen Felder müssen positiv beantwortet sein
<b>ohne Filterfeld</b>	
EF	Einfachauswahl, genau ein Feld muss positiv beantwortet sein
MF_OPTIONAL	Mehrfachauswahl, alle Felder können positiv beantwortet sein
MF_MINDESTENS1	Mehrfachauswahl, mindestens ein Feld muss positiv beantwortet sein
UND	Einfache Regel mit Und-Verknüpfungen

In der Tabelle `BogenFeld` sind abhängige Datenfelder einer Feldgruppe immer als Kann-Felder definiert. Nach Abhängigkeit der Feldgruppenlogik können/müssen diese Felder leer bleiben oder zwingend ausgefüllt werden. Im letztgenannten Fall können die Datenfelder auch als bedingte Muss-Felder bezeichnet werden.

Die Muss- oder Kann-Definition der Datenfelder (Bogen- und Ersatzfelder) im Exportformat unterliegt ebenfalls der Feldgruppenlogik. Ist die Berechnung eines Ersatzfeldes von bedingten Datenfeldern abhängig, so gilt die Feldgruppenlogik auch für diese Ersatzfelder. Wenn die bedingten Datenfelder zwingend ausgefüllt werden müssen, so muss auch das Ersatzfeld zwingend berechnet bzw. exportiert werden.<sup>33</sup>

### Struktur der Tabellen `FeldGruppe` und `FeldgruppeFelder`

Die Feldgruppen sind in den Tabellen `FeldGruppe` und `FeldgruppeFelder` definiert. In der Tabelle `FeldGruppe` (Tabelle 25) sind Name, Typ und die Zuordnung zu einem Modul definiert. Die Verknüpfungstabelle `FeldgruppeFelder` (Tabelle 26) definiert die abhängigen Bogenfelder. Zusätzlich wird hier festgelegt, welche Bogenfelder der Feldgruppe als Filterfeld dienen.

Tabelle 25: Struktur der Tabelle `FeldGruppe`

Feldname	Datentyp	Bemerkung
<code>idFeldGruppe</code>	INTEGER	Primärschlüssel

<sup>33</sup> Die Funktion `verkettentmt` verkettet (zwei oder mehrere) Zeichenfolgen zu einer Zeichenfolge. Hierbei müssen nicht alle im Attribut `Ersatzfeld.formel` aufgeführten Datenfelder ausgefüllt sein. Ist lediglich eine übergebene Zeichenfolge nicht leer, wird diese ohne Verkettung zurückgeliefert.

Feldname	Datentyp	Bemerkung
name	TEXT (64)	Technischer Name der Feldgruppe
fkModul	INTEGER	Obligatorischer Fremdschlüssel zu einem Modul
fkFeldgruppenTyp	INTEGER	Obligatorischer Fremdschlüssel zu einem Feldgruppen- typ
hinweis	TEXT	Bei Filter-Feldgruppen relevant für die Gestaltung der Eingabemaske.  Der Hinweistext informiert den Anwender über die Bedingungen, welche das Ausfüllen von ein oder mehreren abhängigen Feldern erforderlich machen. Der Hinweistext kann bei der Erstellung der Eingabemasken verwendet werden.
fkFilterFeldTyp	CHAR (1)	Attribut wird bei Feldgruppen mit mehreren Filterfeldern gesetzt:  O = Oder-Verknüpfung der positiven Filterbedingungen U = Und-Verknüpfung der positiven Filterbedingungen
fkRegelTyp	CHAR (1)	Fremdschlüssel zur Tabelle RegelTyp:  Die Regeltypen sind die in Abschnitt 9.3 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D  Die generierten Einzelregeln der Feldgruppe haben den gleichen Regeltyp.
nurPositiv	BOOLEAN	Wenn WAHR, dann umfasst die Feldgruppe nur diejenigen Regeln, welche sich auf die positive (Filter)bedingung beziehen.
grauWennNegativ	BOOLEAN	Definiert eine Layout-Feldgruppe, wenn WAHR (siehe unten S. 109, Layout-Feldgruppen)

Tabelle 26: Struktur der Tabelle *FeldgruppeFelder*

Feldname	Datentyp	Bemerkung
idFeldgruppeFelder	INTEGER	Primärschlüssel
fkFeldGruppe	INTEGER	Obligatorischer Fremdschlüssel zur Feldgruppe
fkBogenFeld	INTEGER	Obligatorischer Fremdschlüssel zum Bogenfeld
Bedingung	TEXT	Positive Bedingung für das jeweilige Bogenfeld

Feldname	Datentyp	Bemerkung
istFilter	BOOL	Legt fest, ob das jeweilige Bogenfeld ein Filterfeld ist
bezeichnungSchluesselListe	TEXT	Abkürzende Bezeichnung für eine SchlüsselListe in der Bedingung, wird beim Generieren von Fehlermeldungen verwendet.
tdsListe	BOOLEAN	Das Bogenfeld wird in Regeln als TDS-Listefeld (Abschnitt 9.3.6) verwendet (Vorstellen des @-Zeichens vor Feldnamen).

### Syntax der Feldgruppenregeln

In den Tabellen `FeldGruppe` bzw. `FeldgruppeFelder` sind die positiven Bedingungen für das Filterbogenfeld bzw. die abhängigen Bogenfelder einer Feldgruppe definiert. Jede Bedingung hat folgenden Aufbau:

`<Operator> <Operand>`

Der linke Operand wird hier weggelassen, weil er immer der Name des jeweiligen Bogenfeldes ist. Die komplette Bedingung für das Bogenfeld einer Feldgruppe lautet also:

`<Bogenfeld> <Operator> <Operand>`

Als Operator kann jeder dyadische Operator der Tabelle 23 verwendet werden. Die auf der rechten Seite erlaubten Operanden sind nachfolgend aufgelistet:

- Literale (Tabelle 22)
- LEER
- Kodelisten, in denen auch die Codes eines Schlüssels referenziert werden können
- ICD-Listen oder OPS-Listen



#### Hinweis

Der rechte Operand darf kein Bogenfeld sein, da sich eine Feldbedingung immer genau auf ein Bogenfeld bezieht.

Im Folgenden sind einige mögliche Bedingungen von Feldgruppen dargestellt:

### Formale Definition von Feldgruppen

A sei ein Bogenfeld in einer Feldgruppe. Dann seien  $p(A)$  die positiven und  $n(A)$  die negativen Bedingungen, welche jeweils das Ergebnis wahr oder falsch haben können.

Eine Feldgruppe kann ggf. ein Filterfeld haben, das mit  $F$  bezeichnet wird. Eine Feldgruppe lässt sich dann in folgender Tabelle darstellen:

Tabelle 27: Formale Definition einer Feldgruppe

Feld	Positive Bedingung	Negative Bedingung	Bemerkung
F	$p(F)$	$n(F)$	falls Feldgruppentyp mit Filter
A1	$p(A1)$	$n(A1)$	
A2	$p(A2)$	$n(A2)$	
A3	$p(A3)$	$n(A3)$	
...			
An	$p(An)$	$n(An)$	

Eine Feldgruppe besteht insgesamt aus  $n$  abhängigen Bogenfeldern:

$A1, A2, \dots, An$

In Abhängigkeit von den Feldgruppentypen werden unterschiedliche Einzelregeln generiert.

### Feldgruppen mit Filter

- Regeln der Feldgruppe „Optionale Mehrfachauswahl mit Filterfeld“

$(MF\_OPTIONAL\_FILTER)$

$n(F) \text{ UND } p(A_i) \quad i = 1, \dots, n$

Insgesamt sind  $n$  Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Obligatorische Mehrfachauswahl mit Filterfeld“

$(MF\_MINDESTENS1\_FILTER)$

$n(F) \text{ UND } p(A_i) \quad i = 1, \dots, n$

$p(F) \text{ UND } n(A1) \text{ UND } n(A2) \text{ UND } \dots \text{ UND } n(An)$

Insgesamt sind  $n+1$  Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Mehrfachauswahl mit Filterfeld, alle abhängigen Felder müssen positiv beantwortet sein“

$(MF\_ALLES\_FILTER)$

$n(F) \text{ UND } p(A_i) \quad i = 1, \dots, n$

$p(F) \text{ UND } n(A_i) \quad i = 1, \dots, n$

Insgesamt sind  $2n$  Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Einfachauswahl mit Filter“

$(EF\_FILTER)$

$n(F) \text{ UND } p(A_i) \quad i = 1, \dots, n$

$p(F) \text{ UND } n(A1) \text{ UND } n(A2) \text{ UND } \dots \text{ UND } n(An)$

$p(F) \text{ UND } p(A_j) \text{ UND } p(A_i) \quad \text{für alle unterschiedlichen } i, j = 1, \dots, n$

Insgesamt sind  $n(n+1)/2+1$  Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Optionale Einfachauswahl mit Filter“ (EF\_OPTIONAL\_FILTER)

$$n(F) \text{ UND } p(A_i) \quad i=1,..,n$$

$$p(F) \text{ UND } p(A_j) \text{ UND } p(A_i) \quad \text{für alle unterschiedlichen } i,j=1,..,n$$

Insgesamt sind  $n(n+1)/2$  Einzelregeln mit der Feldgruppe verknüpft.

### Feldgruppen mit Filter: Attribut `nurPositiv`

Wenn in einer Feldgruppe mit Filter das Attribut `nurPositiv` gesetzt ist, so sind nur die Einzelregeln mit positiver Filterbedingung Bestandteil der Feldgruppe.

---

#### Beispiel:

Die Feldgruppe EF\_FILTER mit `nurPositiv=ja` hat folgende Einzelregeln:

$$p(F) \text{ UND } n(A_1) \text{ UND } n(A_2) \text{ UND } \dots \text{ UND } n(A_n)$$

$$p(F) \text{ UND } p(A_j) \text{ UND } p(A_i) \quad \text{für alle unterschiedlichen } i,j=1,..,n$$


---

### Feldgruppen ohne Filter

- Regeln der Feldgruppe „Einfachauswahl“ (EF)

$$n(A_1) \text{ UND } n(A_2) \text{ UND } \dots \text{ UND } n(A_n)$$

$$p(A_j) \text{ UND } p(A_i) \quad \text{für alle unterschiedlichen } i,j=1,..,n$$

Insgesamt sind  $n(n-1)/2+1$  Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Obligatorische Mehrfachauswahl“ (MF\_MINDESTENS1)

$$n(A_1) \text{ UND } n(A_2) \text{ UND } \dots \text{ UND } n(A_n)$$

Insgesamt ist eine Einzelregel mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Und-Regel“ (UND)

$$p(A_1) \text{ UND } p(A_2) \text{ UND } \dots \text{ UND } p(A_n)$$

Insgesamt ist eine Einzelregel mit der Feldgruppe verknüpft.

### Feldgruppen mit mehreren Filterfeldern

Es besteht die Möglichkeit, Feldgruppen mit mehr als einem Filterfeld zu definieren:

Formal gibt es dann die Filterfelder  $F_1, F_2, \dots, F_n$  mit den positiven bzw. negativen Bedingungen  $p(F_j)$  bzw.  $n(F_j)$ . Für alle Filterfelder wird eine positive Bedingung  $p(F_1, \dots, F_n)$  und eine negative Bedingung  $n(F_1, \dots, F_n)$  gebildet. Diese modifizierten Filterbedingungen ersetzen die im Abschnitt 9.3.8 definierten Filterbedingungen  $p(F)$  und  $n(F)$  bei den Einzelregeln.



Die Filterfelder können entweder über eine ODER-Verknüpfung oder eine UND-Verknüpfung miteinander verbunden sein:

$$p(F1, \dots, Fn) = p(F1) \text{ ODER } p(F2) \text{ ODER } \dots \text{ ODER } p(Fn)$$

(ODER-Verknüpfung)

$$p(F1, \dots, Fn) = p(F1) \text{ UND } p(F2) \text{ UND } \dots \text{ UND } p(Fn)$$

(UND-Verknüpfung)

Der Verknüpfungstyp ist im Attribut `fkFilterFeldTyp` der Tabelle `FeldGruppe` hinterlegt.

### Layout-Feldgruppen

Feldgruppen, bei denen das Attribut `grauWennNegativ` in der Datenbanktabelle `FeldGruppe` WAHR ist, werden nachfolgend als Layout-Feldgruppen bezeichnet. Der Attributname `grauWennNegativ` wurde gewählt, weil die abhängigen Felder der Layout-Feldgruppen auf den generierten Dokumentationsbögen ausgegraut sind.

Layout-Feldgruppen haben folgende Eigenschaften:

- Sie haben mindestens ein Filterfeld.
- Jedes abhängige Feld hat die Bedingung `<> LEER` oder `EINSNICHTIN (LEER)` (Attribut `bedingung` in Tabelle `FeldGruppeFelder`).
- Das Attribut `nurPositiv` hat den Wert `FALSCH`.

### 9.3.9 Prüfung von Feldeigenschaften

Die in diesem Abschnitt behandelten feldbezogenen Prüfungen ergeben sich direkt aus den Tabellen `Feld` (bzw. `ErsatzFeld` oder `ZusatzFeld`) und `BogenFeld` (bzw. `ExportFormat`) und werden vor Evaluation der feldübergreifenden Regeln durchgeführt.

Die hier beschriebenen Prüfungen sind nur in Form von Feldeigenschaften – nicht aber in Regelsyntax – in der Datenbank für QS-Dokumentation hinterlegt.

### Überprüfung des Formats

Die Formatprüfung bezieht sich auf den Exportdatensatz (Abschnitt B 9.5.2): Die QS-Dokumentations- bzw. Exportsoftware muss Daten im korrekten Format generieren, was durch die datenentgegennehmenden Stellen geprüft wird. Für Exportfelder (Tabelle `ExportFormat`), die einen Bezug zu einem Ersatzfeld (Tabelle `ErsatzFeld`) bzw. zu einem Zusatzfeld (Tabelle `ZusatzFeld`) haben, sind die standardisierten Fehlertexte anzupassen.

Die Prüfung bezieht sich insbesondere darauf, ob der Feldinhalt dem in der Spezifikation definierten Basistyp entspricht. Beispielsweise sind Buchstaben beim Basistyp `GANZEZAHL` nicht erlaubt.

#### Standardisierter Fehlertext für Formatfehler eines Ersatzfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Ersatzfeld.name> '<ErsatzFeld.bezeichnung>' ist kein gültiger <BasisTyp.name> Wert (<BasisTyp.bezeichnung> <BasisTyp.format>).

#### Standardisierter Fehlertext für Formatfehler eines Bogenfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Feld.name> '<BogenFeld.bezeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) ist kein gültiger <BasisTyp.name> Wert (<BasisTyp.bezeichnung> <BasisTyp.format>).

#### Standardisierter Fehlertext für Muss-Fehler eines Zusatzfeldes

Das Zusatzfeld <Modul.name>:<Bogen.name>:<ZusatzFeld.name> '<ZusatzFeld.bezeichnung>' ist kein gültiger <BasisTyp.name> Wert (<BasisTyp.bezeichnung> <BasisTyp.format>).

#### Überprüfung der Feldlänge

Die Feldlängenprüfung bezieht sich darauf, ob die Anzahl der Zeichen eines Wertes die spezifizierte Länge<sup>34</sup> (Attribut *laenge* in Tabelle *Feld*) des Feldes überschreitet.

#### Standardisierter Fehlertext für Längenfehler eines Ersatzfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<ErsatzFeld.name> '<ErsatzFeld.bezeichnung>' überschreitet die zulässige Feldlänge <Feld.laenge>.

#### Standardisierter Fehlertext für Längenfehler eines Bogenfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Feld.name> '<BogenFeld.bezeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) überschreitet die zulässige Feldlänge <Feld.laenge>.

#### Standardisierter Fehlertext für Fehler eines Zusatzfeldes

Das Zusatzfeld <Modul.name>:<Bogen.name>:<ZusatzFeld.name> '<ZusatzFeld.bezeichnung>' überschreitet die zulässige Feldlänge <ZusatzFeld.laenge>.

### **Überprüfung der Schlüsselcodes**

Die Überprüfung von Schlüsselcodes bezieht sich darauf, ob bei Schlüsselfeldern nur zulässige Schlüsselcodes verwendet werden.

---

<sup>34</sup> Wenn bei einem Ersatz die Länge nicht spezifiziert ist, so entfällt die Prüfung.

#### Standardisierter Fehlertext bei unzulässigen Schlüsselcodes eines Ersatzfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<ErsatzFeld.name> '<ErsatzFeld.bezeichnung>' ist kein zulässiger Code des Schlüssels <Schluessel.name> (<Schluessel.bezeichnung>).

#### Standardisierter Fehlertext bei unzulässigen Schlüsselcodes eines Bogenfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Feld.name> '<BogenFeld.bezeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) ist kein zulässiger Code des Schlüssels <Schluessel.name> (<Schluessel.bezeichnung>).

#### Standardisierter Fehlertext für Fehler eines Zusatzfeldes

Das Zusatzfeld <Modul.name>:<Bogen.name>:<ZusatzFeld.name> '<ZusatzFeld.bezeichnung>' ist kein zulässiger Code des Schlüssels <Schluessel.name> (<Schluessel.bezeichnung>).

### **Besonderheiten bei externen Schlüsseln**

- Bei externen DIMDI-Schlüsseln (ICD-10-GM oder OPS) sind die jeweils gültigen amtlichen Kataloge zu verwenden.
- Nicht-terminale ICD- oder OPS-Kodes sind unzulässig!

### **Überprüfung numerischer Wertebereiche**

Bei numerischen Datenfeldern (Typ ZAHL oder GANZEZAHL) ist hart zu überprüfen, ob der Zahlenwert im durch die Attribute min und max (Tabelle FELD) definierten Wertebereich liegt:

- wert < min (nur prüfen, wenn min <> LEER)
- wert > max (nur prüfen, wenn max <> LEER)

In der Tabelle Feld sind weitere Unter- und Obergrenzen (Attribute min/max und minWeich/maxWeich) für Prüfungen definiert. In wenigen Fällen sind auch in der Tabelle BogenFeld Grenzen definiert. Falls vorhanden (= not null), werden die in der Tabelle BogenFeld gesetzten speziellen Wertgrenzen statt der Grenzen in der Tabelle Feld angewandt.

Eine Übersicht über die in numerischen Datenfeldern definierten harten und weichen Wertebereiche bietet die Abfrage WertebereicheNumerischerFelder.

#### Standardisierter Fehlertext bei Unterschreitung einer Wertebereichsgrenze eines Ersatzfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<ErsatzFeld.name> '<ErsatzFeld.bezeichnung>' ist kleiner als '<Feld.min>'.

#### Standardisierter Fehlertext bei Unterschreitung einer Wertebereichsgrenze eines Bogenfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Feld.name> '<BogenFeld.bezeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) ist kleiner als '<Feld.min>'.

#### Standardisierter Fehlertext bei Überschreitung einer Wertebereichsgrenze eines Ersatzfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<ErsatzFeld.name> '<ErsatzFeld.bezeichnung>' ist größer als '<Feld.max>'.

#### Standardisierter Fehlertext bei Überschreitung einer Wertebereichsgrenze eines Bogenfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Feld.name> '<BogenFeld.bezeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) ist größer als '<Feld.max>'.

Bei weichen Plausibilitätsverletzungen ist dem Fehlertext das Wort „Hinweis“ voranzustellen.

### **Überprüfung der Muss-Felder**

Ein nicht ausgefülltes Muss-Feld führt zu einer Regelverletzung.

#### Standardisierter Fehlertext für Muss-Fehler eines Ersatzfeldes

Das Datenfeld '<Modul.name>:<Bogen.name>:<ErsatzFeld.name>' '<ErsatzFeld.bezeichnung>' muss einen gültigen Wert enthalten.

#### Standardisierter Fehlertext für Muss-Fehler eines Bogenfeldes

Das Datenfeld '<Modul.name>:<Bogen.name>:<Feld.name>' '<BogenFeld.bezeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) muss einen gültigen Wert enthalten.

#### Standardisierter Fehlertext für Muss-Fehler eines Zusatzfeldes

Das Zusatzfeld <Modul.name>:<Bogen.name>:<ZusatzFeld.name> '<ZusatzFeld.bezeichnung>' muss einen gültigen Wert enthalten.

### **9.3.10 Verfahren für die Evaluation von Regeln**

Jedes Bogenfeld – jedes Listefeld – darf keine Regel aus der Tabelle QSDOK.Regeln verletzen. Nach der Dokumentation muss jedes Exportfeld die QSDOK.Regeln mit `guelutigNachExport` sowie die referenzierten Mehrfachregeln einhalten.

Grundsätzlich muss jede gem. Abschnitt 9.3.2 formulierte Regel evaluiert (ausgeführt) werden, wenn keine der folgenden Bedingungen zutrifft:

1. Mindestens ein referenziertes Bogenfeld ist auf einem nicht vorhandenen Teildatensatz.

2. Für mindestens ein referenziertes Bogenfeld<sup>35</sup> schlägt eine harte Feldprüfung (Abschnitt B 9.3.9) fehl.<sup>36</sup>
3. Ein Feld der Regel ist nicht ausgefüllt (LEER) und **keine** der folgenden Teilbedingung trifft in Bezug auf das leere Feld zu:
  - Es ist in einer Liste enthalten, die mit einem Listenoperator (EINSIN, KEINSIN, JEDESIN, EINSNICHTIN) geprüft wird bzw. wird direkt gegen eine Liste geprüft (IN, NICHTIN).
  - Es wird in der Regel explizit mit <> oder = auf LEER geprüft.
  - Jeder Operation auf einen Wert <> LEER ist eine ODER-Verknüpfte Prüfung auf LEER direkt vorgeschaltet (Feld = LEER ODER Feld Operator Operand).
4. Eine Funktion der Regel hat das Ergebnis LEER und wird in der Regel nicht explizit mit <> oder = auf LEER geprüft. (Hinweis: alle Argumente der Funktionen müssen dennoch/immer ungleich LEER sein.)

### Teildatensatzübergreifende Regeln

Teildatensatzübergreifende Regeln (Abschnitt 9.3.6) müssen u.U. mehrfach evaluiert werden (für jede Kombination von Teildatensätzen, die von der Regel betroffen ist).



#### Hinweis

In wenigen Einzelfällen beziehen sich Plausibilitätsregeln auf mehr als zwei Teildatensätze.

---

## 9.4 Listen von Schlüsselkodes (OPS;ICD)

In der Spezifikation sind Listen von ICD- und OPS-Kodes in separaten Tabellen definiert. Jede Liste hat einen technischen Namen und eine erläuternde Bezeichnung. Die Listen sind in insgesamt vier Tabellen der Spezifikationsdatenbank definiert. Die technischen Namen und erläuternden Bezeichnungen sind in den Tabellen OPSListe und ICDListe definiert. Die Kodes finden sich in den Tabellen OPSWert und ICDWert.

### 9.4.1 OPS-Listen

Jede OPS-Liste ist charakterisiert durch ihren Namen (Attribut name in Tabelle OPSListe), welcher nach folgendem Schema gebildet wird:

{<TEXT>\_}OPS{\_<TEXT>}

Hinter <TEXT> verbirgt sich ein frei wählbarer Name (Erlaubte Zeichen: A-Z, a-z, 0-9, .. Umlaute sind nicht erlaubt.). Die {}-Ausdrücke sind optional.

---

<sup>35</sup> Die Verbindung zwischen Regeln und Bogenfeldern geschieht über die Tabelle RegelFelder, siehe Regelfelder (Bogenfelder einer Regel), Abschnitt B 9.3.1.

<sup>36</sup> Erst bei Fehlerfreiheit der feldbezogenen Prüfungen werden die feldübergreifenden Prüfungen durchgeführt.

**Beispiel aus PPP:**

OPS\_RA\_Gueltigkeit

OPS-Kodes der Regelaufgaben bei der die Gültigkeit verpflichtend zu dokumentieren ist

Tabelle 28: Identitätsprüfung zwischen dokumentierten OPS-Kodes und Kodes von OPS-Listen

Dokumentierter OPS-Kode	OPS-Kode der OPS-Liste	Bedingung für Gleichheit (= ist Stringvergleich)
normCodeDok + seiteDok	normCodeListe	normCodeDok = normCodeListe
normCodeDok	normCodeListe	normCodeDok = normCodeListe

Bei allen Prüfungen mit OPS-Listen sind diese Regeln zu beachten.

### 9.4.2 ICD-Listen

Jede ICD-Liste ist charakterisiert durch ihren Namen (Attribut `name` in Tabelle `ICDListe`), welcher per definitionem folgendem Schema folgt:

$$\{ \langle \text{TEXT} \rangle \_ \} \text{ICD} \{ \_ \langle \text{TEXT} \rangle \}$$

Hinter `<TEXT>` verbirgt sich ein frei wählbarer Name (Erlaubte Zeichen: A-Z, a-z, 0-9, \_, Umlaute sind nicht erlaubt).

Die in der Tabelle `ICDWert` (Attribut `code`) definierten Kodes entsprechen der Systematik der Spalte `NormCode` aus Tabelle `Codes` in den Katalogen des DIMDI:

Der ICD-10-GM wird 4- oder 5-stellig kodiert, kann aber durch ein Suffix bestehend aus `[A|V|Z][L|R|B]` (ohne Leerzeichen, z. B. „K41.9ZL“) ergänzt werden.

**Hinweis**

Die Suffixe \*, +, ! entfallen in der Spezifikation!

Es ist zu beachten, dass im Krankenhaus dokumentierte ICD-Kodes die Suffixe \*, +, ! enthalten können.

### 9.4.3 Relationstabellen ICD- und OPS-Listen

Für die ICD- und OPS-Listen wird eine Referenz zum Attribut `fkBogenfeld` hergestellt. Um diesen Zusammenhang in der Datenbank abzubilden, wurden die neuen Tabellen `ICDRelation` und `OPSRelation` aufgenommen.

Über diese neuen Tabellen kann eine weitere Anforderung an die Spezifikation abgebildet werden, nämlich die für Plausibilitätsregeln notwendigen „Gesamtlisten“.

Um den Zusammenhang von Bogenfeld, Einzellisten und Gesamtlisten in der Datenbank abzubilden, werden die Tabellen `ICDRelation` und `OPSRelation` genutzt.

IDRelation_C	fkOPSListe	fkGesamtliste
2	OPS_RA	OPS_Gesamt
3	OPS_RA_Gultigkeit	OPS_Gesamt

Abbildung 7: Tabelle OPSRelation

## 9.5 Exportfeldbeschreibung

Neben der Datenfeldbeschreibung (Abschnitt 9.2) enthält die Spezifikationsdatenbank die Beschreibung der Exportfelder für ein Modul (Exportdatensatz). Diese werden zum Teil über Ersatzfelder berechnet. Es wird zudem zwischen Dokumentationsmodulen (Abschnitt 9.2.1) und Exportmodulen unterschieden.

### 9.5.1 Exportmodule

Um die unterschiedlichen Datenflüsse mit unterschiedlichen Datenannahmestellen für die Datenübermittlung zu berücksichtigen, wird zwischen den Dokumentationsmodulen (Tabelle `Modul`) und den Exportmodulen (Tabelle `Exportmodul`) unterschieden.

Tabelle 29: Struktur der Tabelle `ExportModul`

Feldname	Datentyp	Bemerkung
<code>idExportModul</code>	INTEGER	Primärschlüssel
<code>fkModul</code>	INTEGER	Bezug zum Dokumentationsmodul
<code>Name</code>	TEXT	Technischer Name (muss eindeutig sein)
<code>bezeichnung</code>	TEXT	Bezeichnung des Exportmoduls
<code>existenzBedingung</code>	TEXT	Definiert, unter welcher Bedingung das Modul in ein definiertes Exportmodul transformiert wird.
<code>type_QS_data</code>	TEXT	Datentyp im XML-Schema
<code>ersatzBedingungMDS</code>	TEXT	Definiert, unter welcher Bedingung das Modul MDS in ein definiertes Exportmodul transformiert wird. In QSFFx kommt kein MDS zur Anwendung.
<code>type_QS_data_mds</code>	TEXT	Datentyp im XML-Schema
<code>fkExportzeitraumEntleJ</code>	INTEGER	Definiert den Exportzeitraum für ein Exportmodul. Die hinterlegten Zeitpunkte sind in der Tabelle <code>Exportzeitraum</code> definiert.

Feldname	Datentyp	Bemerkung
		Exportzeitraum.exportBis = Datenlieferfrist inkl. Korrekturfrist
fkExportzeitraumEntleJ1	INTEGER	Definiert den Exportzeitraum für ein Exportmodul im Spezifikationsjahr+ 1. Die hinterlegten Zeitpunkte sind in der Tabelle Exportzeitraum definiert. Exportzeitraum.exportBis = Datenlieferfrist inkl. Korrekturfrist
pid	BOOLEAN	Handelt es sich um ein Modul zur Follow-up-Erhebung?
vpb	BOOLEAN	Handelt es sich um ein Modul zur Befragung von Patienten?
fkPseudonymVerfahren	INTEGER	Pseudonymisierung von FU-Verfahren

Das Attribut `type_QS_data` gibt Auskunft darüber, welchem Datentyp dies im XML-Schema entspricht.

---

#### Beispiel:

Beim Exportmodul `F_NW` ist der Datentyp im XML-Schema `qs_data_fnw_type`.

---

Softwareanbietern soll hiermit die Integration eines Mechanismus der automatischen Datentypzuweisung ermöglicht werden, um den Aufwand zu reduzieren und Fehler zu vermeiden.

### 9.5.2 Exportdatensatz

Der Exportdatensatz enthält die Exportfelder für ein Modul. Welche Zusatzfelder, Bogenfelder und/oder Ersatzfelder den Exportdatensatz pro Modul bilden, ist in Tabelle `ExportFormat` definiert.

#### Zusatzfelder<sup>37</sup>

Ein Exportfelddatensatz beginnt immer mit den folgenden Zusatzfeldern:

Vorgangsnr	=	Vorgangsnummer
VorgangsnrGuid	=	Vorgangsnummer (GUID)
VersionNr	=	Versionsnummer
Storno	=	Stornierung eines Datensatzes (inkl. aller Teildatensätze)
Modul	=	Bezeichnung des Exportmoduls

---

<sup>37</sup> Dabei ist zu beachten, dass die Feldnamen beim Export durch die XML-Elemente zu ersetzen sind.



Bogen = Teildatensatz (Bogen)  
DokAbschlDat = Dokumentationsabschlussdatum

Ein neuer Teildatensatz beginnt mit den Zusatzfeldern `Vorgangsnr`, `VorgangsnrGuid` und `VersionNr`. Teildatensätze mit einem definierten Mutterteildatensatz beinhalten zusätzlich das Zusatzfeld `IdBogenFeldMutter` (= Wert des eindeutigen Bogenfeldes des Mutterteildatensatzes).

Zusatzfelder, welche nicht in der Datenfeldbeschreibung (Tabelle `BogenFeld`) eines Moduls enthalten sind, werden von der QS-Dokumentationssoftware ausgefüllt.<sup>38</sup>

Einige der in der Tabelle `ZusatzFeld` definierten Zusatzfelder werden nachfolgend erläutert:

- Das übertragene Speicherdatum `DokAbschlDat` (Datum des Dokumentationsabschlusses bzw. der Freigabe des Datensatzes für den Export) ist nicht Teil der Datenbank für Auswertungen und wird nur für organisatorische Zwecke verwendet. Das `DokAbschlDat` ist das Datum der letzten Änderung des gesamten Datensatzes.
- Die Versionsnummer (`VersionNr`) gibt an, welche Version des Datensatzes übertragen wird.

In der Regel wird die Versionsnummer 1 lauten, d.h., dass der nach dem ersten Dokumentationsabschluss freigegebene Datensatz übertragen wird. Muss ein korrigierter Datensatz erneut eingesandt werden, so muss die Versionsnummer vom dokumentierenden System um eins erhöht werden. Die neue Version des Datensatzes wird bei der Entgegennahme geprüft und überschreibt bei Korrektheit die alte Version des Datensatzes.



#### Achtung

Wenn die entgegennehmende Stelle einen Datensatz mit derselben Versionsnummer ein zweites Mal erhält, so wird dieser zurückgewiesen.

---

- Der Eintrag 1 im Zusatzfeld `Storno` veranlasst die datenentgegennehmende Stelle, den übermittelten Datensatz einschließlich seiner Vorversion(en) als „storniert“ zu kennzeichnen.
- Das Zusatzfeld `IdBogenFeldMutter` wird bei Teildatensätzen eingefügt, welche einen mehrfach anlegbaren Elternteildatensatz (Attribut `fkBogenZahl` = '\*' oder '+') haben. In diesem Fall wird die identifizierende Nummer des Elternteildatensatzes (konfiguriert über `Bogen.fkEindeutigBogenFeld`) im Kind-Teildatensatz übermittelt.
- Eine vollständige Liste der möglichen Zusatzfelder findet sich in der Tabelle `ZusatzFeld` der Spezifikationsdatenbank zur QS-Dokumentation. Zusatzfelder sind in Tabelle 30 definiert.

---

<sup>38</sup> Hier gilt also nicht der Grundsatz, dass Felder nicht vorbelegt sein dürfen.

Tabelle 30. Struktur der Tabelle *ZusatzFeld*

Feldname	Datentyp	Bemerkung
idZusatzFeld	INTEGER	Primärschlüssel
name	TEXT	Technischer Name (muss eindeutig sein)
bezeichnung	TEXT	Bezeichnung des Zusatzfeldes
fkBasisTyp	INTEGER	Fremdschlüssel zur Tabelle <i>BasisTyp</i>
fkSchluessel	INTEGER	Fremdschlüssel zur Tabelle <i>Schlüssel</i>
laenge	INTEGER	Feldlänge des Zusatzfeldes
nachKommaLaenge	INTEGER	Anzahl der Nachkommastellen
istListe	BOOLEAN	Wenn <code>istListe = WAHR</code> , so sind die vom betreffenden Feld abgeleiteten Bogenfelder Listenfelder.
min	INTEGER	Harte Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulübergreifend). Die Definition ist optional.
max	INTEGER	Harte Obergrenze des Wertebereichs eines numerischen Datenfeldes (modulübergreifend). Die Definition ist optional.
sortierNr	INTEGER	Sortiernummer
nurBasisTDS	BOOLEAN	Wenn <code>nurBasisTDS = WAHR</code> , so ist das Zusatzfeld im Exortdatensatz nur im Basisbogen enthalten
nurKreuzSternKindTDS	BOOLEAN	Wenn <code>nurKreuzSternKindTDS = WAHR</code> , so ist das Zusatzfeld nur bei Teildatensätzen mit einem definierten Mutterteildatensatz enthalten

Die Exportfelder werden pro Exportmodul exportiert. Hierfür erhält das Zusatzfeld `Modul` den Datentyp `SCHLUESSEL` und lässt nur die im Schlüssel `Exportmodul` definierten Codes zu.



#### Hinweis

Bei den Zusatzfeldern ist zu beachten, dass die Feldnamen beim Export durch die entsprechenden XML-Elemente zu ersetzen sind (Abschnitt 9.7.1).

### Ersatzfelder für den Datenexport

Ersatzfelder werden aus einem oder mehreren Feldern der Datenfeldbeschreibung berechnet. Mit einem Ersatzfeld verknüpfte Bogenfelder werden nicht exportiert, wenn sie nicht als `<bleibt>` gekennzeichnet sind. Stattdessen werden ein oder mehrere Ersatzfelder exportiert. Vorrangig

dienen Ersatzfelder der Anonymisierung beim Datenexport. Die verwendeten Ersatzfelder sind in der Tabelle `ErsatzFeld` gespeichert.

Tabelle 31. Struktur der Tabelle *Ersatzfeld*

Feldname	Datentyp	Bemerkung
<code>idErsatzFeld</code>	INTEGER	Primärschlüssel
<code>name</code>	TEXT	Technischer Name (muss eindeutig sein)
<code>bezeichnung</code>	TEXT	Bezeichnung des Ersatzfeldes
<code>laenge</code>	INTEGER	maximale Zeichenlänge (in Verbindung mit <code>BasisTyp</code> ZAHL inklusive Punkt und <code>nachKommaLaenge</code> )
<code>nachKommaLaenge</code>	INTEGER	Anzahl der Nachkommastellen
<code>formel</code>	TEXT	Berechnungsformel der Ersatzfelder
<code>fkBasisTyp</code>	INTEGER	Obligatorischer Fremdschlüssel zum <code>BasisTyp</code>
<code>fkSchluessel</code>	INTEGER	Optionalen Fremdschlüssel zum Schlüssel
<code>istListe</code>	BOOLEAN	Listenfeld
<code>min</code>	DOUBLE	Harte Untergrenze des Wertebereichs
<code>max</code>	DOUBLE	Harte Obergrenze des Wertebereichs
<code>formatAnweisung</code>	KURZER TEXT	Regulärer Ausdruck für die Formatprüfung (z. B. <code>[0-9]{9}</code> )

### Ersatzfelder, die nicht berechnet werden können

Es kann auch vorkommen, dass Ersatzfelder für einen Datensatz nicht berechnet werden können, weil die der Berechnung zugrundeliegenden Bogenfelder nicht ausgefüllt (`LEER`) sind. Folgende allgemeine Regeln gelten für die Berechnung von Ersatzfeldern:

- Wenn die Bogenfelder, aus denen ein Ersatzfeld berechnet wird, dokumentiert sind (`<> LEER`), so ist das entsprechende Ersatzfeld zu berechnen und zu exportieren.
- Wenn eines der beteiligten Bogenfelder nicht ausgefüllt ist und somit auch kein Ersatzfeld berechnet werden kann, wird kein Wert für das Ersatzfeld exportiert (bleibt `LEER`). Eine Ausnahme bildet die Funktion `verkettentmt`. Die Funktion verkettet (zwei oder mehrere) Zeichenfolgen zu einer Zeichenfolge. Ist lediglich eine übergebene Zeichenfolge nicht leer, wird diese ohne Verkettung zurückgeliefert.

### Anonymisierungsvorschriften

Die Anonymisierung von Datenfeldern wird wie aus der folgenden Tabelle 32 Struktur der Tabelle `ErsatzFuerFeld` ersichtlich konfiguriert. Für die Programmierung der Exportfelder ist dieser Abschnitt nicht relevant, da die Exportfelder direkt über die Abfrage `ExportFelderFürEin-`

Modul bzw. die Tabelle `ExportFormat` ermittelt werden können. Die Tabelle `ErsatzFuerFeld` ordnet einem Feld (Tabelle `Feld`) oder Bogenfeld ein oder mehrere Ersatzfelder zu. Die über das Attribut `fkFeld` definierte Anonymisierung ist die Standardanonymisierung für alle Module. Sie kann jedoch durch eine modulspezifische Anonymisierung überschrieben werden: Wenn ein Ersatzfeld mit einem Bogenfeld (über Attribut `fkBogenFeld`) verknüpft ist, wird statt des Bogenfelds das berechnete Ersatzfeld in die Exportdatei des Teildatensatzes geschrieben.

Tabelle 32. Struktur der Tabelle `ErsatzFuerFeld`

Feldname	Datentyp	Bemerkung
<code>idErsatzFuerFeld</code>	INTEGER	Primärschlüssel
<code>fkFeld</code>	INTEGER	Optionaler Fremdschlüssel zur Tabelle <code>Feld</code>
<code>fkBogenFeld</code>	TEXT	Optionaler Fremdschlüssel zur Tabelle <code>BogenFeld</code>
<code>fkErsatzFeld</code>	TEXT	Obligatorischer Fremdschlüssel zur Tabelle <code>ErsatzFeld</code>
<code>parametrierbar</code>	BOOL	Kennzeichen, ob das Feld parametrierbar ist

Wenn das Ersatzfeld `<entfällt>` mit einem Bogenfeld verknüpft ist, entfällt das Bogenfeld in der Exportdatei. Wenn das Ersatzfeld `<bleibt>` mit einem Bogenfeld verknüpft ist, wird das Bogenfeld unverändert in die Exportdatei übernommen.

Wenn ein einziges Ersatzfeld sowohl über `fkFeld` als auch über `fkBogenFeld` definiert ist („doppelte Definition“), hat die spezielle Anonymisierung (über `fkBogenFeld`) Vorrang. Die allgemeinen Anonymisierungen (`fkFeld`) werden ignoriert. Die allgemeine Definition kommt nur in den Modulen zur Anwendung, in denen keine spezielle Anonymisierung vorliegt.

### Parametrierung

Die Verknüpfung zwischen `Feld` bzw. `Bogenfeld` und `Ersatzfeld` kann parametriert werden (Attribut `parametrierbar`). Parametrierbare Ersatzfelder erscheinen immer als eigenes Element in der Exportdatei. Es ist aber über die Dokumentationssoftware konfigurierbar, ob die Werte auch tatsächlich exportiert werden. Auf diese Weise können spezifische Erfordernisse zum Datenschutz auf Landesebene berücksichtigt werden.

### Muss-Felder des Exportdatensatzes

Verbindlich für die Muss-/Kann-Prüfung ist die Definition in der Tabelle `BogenFeld`. Die Muss-/Kann-Zuordnungen im Exportdatensatz werden hieraus abgeleitet:

- Für Exportfelder, die nicht pseudonymisiert werden und die keine Listfelder sind, entspricht die Muss/Kann-Zuordnung der Definition in der Tabelle `BogenFeld`.

- Die Muss-/Kann-Zuordnung der pseudonymisierten Datenfelder (Ersatzfelder) ergibt sich logisch aus der Berechnungsformel (Attribut `formel` in Tabelle `ErsatzFeld`). Beispielsweise ist ein Ersatzfeld ein Muss-Feld, wenn alle an der Berechnung beteiligten Bogenfelder Muss-Felder sind.
- Bei Muss-Listenfeldern der Tabelle `BogenFeld` ist grundsätzlich nur das erste Element ein Muss-Feld, die weiteren Elementfelder sind Kann-Felder. Hierbei ist zu beachten, dass Exportfelder für Listenfelder nicht pro Listenelement, sondern pro Listefeld dargestellt werden. Die Anzahl der Elemente von Listenfeldern ist den Abfragen `Exportfelder` und `ExportfelderFürEinModul` zu entnehmen (`ExportfelderFürEinModul.elemente`).
- Zusatzfelder der Tabelle `ExportFormat` sind Muss-Felder, außer wenn sie Schlüsselfelder mit einem `Null`-Schlüssel sind.



#### Achtung

Wenn ein Listefeld als Muss-Feld deklariert ist, so ist nur das erste Exportfeld der Liste ein Muss-Feld, die restlichen Elemente sind Kann-Felder. Wenn ein Listefeld als Kann-Feld deklariert ist, so sind alle exportierten Elemente ebenfalls Kann-Felder.

Das Nichtausführen der erforderlichen Muss-Prüfungen kann gravierende Folgen für die Auswertung haben!

---

Als Hilfestellung für Datenannahmestellen bei der Umsetzung gilt das Attribut `fkMussKann` in der Tabelle `ExportFormat`, deren Inhalte automatisch generiert werden.

## 9.6 Versionierung

Im Folgenden werden die Tabelle `Version`, der Abgleich zu vorherigen Versionen, die Abgrenzung zwischen Spezifikationsjahren und Datensatzformaten sowie die Version von Exportverfahren und -dateien beschrieben.

### 9.6.1 Grundlegende Definitionen

In der Tabelle `Version` finden sich Informationen zur Version der Spezifikationsdatenbank. Die wichtigsten Eigenschaften einer Version sind der Versionsname (Attribut `name`) und die Gültigkeitszeiträume (Attribute `ab` und `bis`). Der Gültigkeitszeitraum einer Version ist in der Regel ein Spezifikationsjahr.

Jedes Modul der Datenbank hat eine Version (vgl. Attribut `fkVersion` in Tabelle `Modul`). In einer Spezifikationsdatenbank haben alle Module dieselbe Version. Diese entspricht immer der in Tabelle `Version` als gültig gekennzeichneten Version. Über die in der Datenbank definierten Relationen sind auch für alle Bogenfelder (Tabelle `BogenFeld`), Exportfelder (Tabelle `ExportFormat`) und Plausibilitätsregeln (Tabelle `Regeln`) Versionen definiert.



#### Hinweis

---

Grundsätzlich baut eine QS-Dokumentationsdatenbank für ein Spezifikationsjahr auf der QS-Dokumentationsdatenbank aus dem vorherigen Spezifikationsjahr auf. Das Delta wird in den Tabellen `DeltaAttribut`, `DeltaGeloescht` und `DeltaNeu` dargestellt. Sollte nach der ersten Veröffentlichung der QS-Dokumentationsdatenbank für ein Spezifikationsjahr noch ein Update der QS-Dokumentationsdatenbank aus dem vorherigen Spezifikationsjahr notwendig werden, dann divergieren beide Datenbanken. Würde man beide Datenbanken kombinieren, dann wären die IDs nicht garantiert rechtseindeutig. Eine `idVersion` könnte dann zum Beispiel auf unterschiedliche Werte in `QSDOK.Version` verweisen.

---

## Status der Spezifikation

Versionen können den Status `in Entwicklung`, `final` oder `Update` der finalen Spezifikation haben. Diese Zustände werden zum Nachschlagen in der Tabelle `VersStatus` verwaltet. Das Attribut `gueltig` zeigt die gültige Version der Datenbank an. Es darf nur eine einzige Version als gültig markiert sein. Hat eine Spezifikationsdatenbank den Status `in Entwicklung`, kann `Modul.fkVersion` als `ungueltig` markierte Versionen enthalten, um Zwischenstände abzubilden.

## Historie der Versionen

Die Tabelle `Version` enthält auch einen Selbstbezug (Attribut `fkVersion`), der die Identifizierung der Vorgängerversion ermöglicht. Die Vorgängerversion der Spezifikation 2023 V01 ist die Version 2023 V01 Alpha.

### 9.6.2 Delta-Informationen zur vorhergehenden Version

Um den Benutzern der Spezifikation umfassende Informationen zu den jeweiligen Änderungen zur Verfügung zu stellen, enthält die Spezifikationsdatenbank Tabellen, die den Änderungsstand im Vergleich mit der letzten gültigen Version der Datenbank anzeigen. Diese sogenannten Delta-Tabellen werden automatisch generiert. Es gibt drei Delta-Tabellen, die die neuen Entitäten, die gelöschten Entitäten und die geänderten Attributwerte weiterbestehender Entitäten aufzeigen.

## Neue Entitäten

Über die Attribute `id` und `fkTabellenStruktur` in der Tabelle `DeltaNeu` (siehe Tabelle 33) ist die Zuordnung zu den Entitäten der Datenbank möglich.

Tabelle 33. Struktur der Tabelle `DeltaNeu`

Feldname	Datentyp	Bemerkung
<code>idDeltaNeu</code>	INTEGER	Primärschlüssel
<code>id</code>	TEXT	ID der Entität, die eingefügt wurde
<code>fkTabellenStruktur</code>	ZAHL	Bezug zur Tabelle, in die die Entität eingefügt wurde

Feldname	Datentyp	Bemerkung
bemerkung	MEMO	Begründung für die Ergänzung

### Geänderte Entitäten

Über die Attribute `id` und `fkTabellenFeldStruktur` der Tabelle `DeltaAttribut` (siehe Tabelle 34) ist die Zuordnung zu den Attributen der Entitäten der Datenbank möglich.

Tabelle 34: Struktur der Tabelle *DeltaAttribut*

Feldname	Datentyp	Bemerkung
<code>idDeltaAttribut</code>	AUTOWERT	Primärschlüssel
<code>id</code>	TEXT	ID der Entität, die geändert wurde
<code>fkTabellenFeldStruktur</code>	ZAHL	Bezug zum Attribut einer Tabelle, in der die Entität geändert wurde
<code>alterInhalt</code>	MEMO	Alter Inhalt der geänderten Entität in der letzten finalen Spezifikation
<code>neuerInhalt</code>	MEMO	Neuer Inhalt dieser Entität in der aktuellen Spezifikation
<code>bemerkung</code>	MEMO	Begründung für die Ergänzung

### Gelöschte Entitäten

Über die Attribute `id` und `fkTabellenStruktur` in der Tabelle `DeltaGeloesch` (siehe Tabelle 35) ist die Zuordnung zu den Entitäten der Datenbank möglich.

Tabelle 35: Struktur der Tabelle *DeltaGeloesch*

Feldname	Datentyp	Bemerkung
<code>idDeltaGeloesch</code>	INTEGER	Primärschlüssel
<code>id</code>	TEXT	ID der Entität, die gelöscht wurde
<code>fkTabellenStruktur</code>	ZAHL	Bezug zum Attribut einer Tabelle, in der die Entität gelöscht wurde
<code>bemerkung</code>	MEMO	Ggf. Bemerkung zur Ergänzung

### Konfiguration der Delta-Berechnung

Es werden nur dann Delta-Informationen zu einer Entität angezeigt, wenn für mindestens ein Attribut der Entität in der Tabelle `TabellenFeldStruktur` das Attribut `deltaAktiv = WAHR` gesetzt ist.

Tabelle 36: Inhalt der Tabelle *TabellenFeldStruktur* (*fkTabellenFeldStruktur* = *Regeln*)

idTabelle	feldName	...	deltaAktiv	fk TabellenFeldStruktur
445	idRegeln		Nein	Regeln
450	fkMehrfachRegel		Ja	Regeln
451	fkFeldGruppe		Ja	Regeln
449	fkRegelTyp		Ja	Regeln
446	bedingung		Ja	Regeln
447	meldung		Ja	Regeln
448	fkModul		Ja	Regeln
454	gueltigNachExport		Ja	Regeln

**Beispiel:**

Das Delta für die Tabelle *Regeln* wird für die Attribute *fkFeldGruppe*, *fkMehrfachRegel*, *fkRegelTyp*, *bedingung*, *meldung*, *fkModul* und *gueltigNachExport* erzeugt.

### 9.6.3 Abgrenzung zwischen Spezifikationsjahren und Datensatzformaten

Die Datenannahmestellen müssen Datensätze von Leistungserbringern entgegennehmen, die in den Gültigkeitszeitraum der Version einer finalen Datenbank fallen.

### 9.6.4 Version des Exportverfahrens

In der Spezifikationsdatenbank für die QS-Dokumentation wird in der Tabelle *Version* neben der gültigen Version (*gueltig*) auch das gültige Exportverfahren (*gueltigExportVerfahren*) angegeben. Dieses Attribut gibt an, welches Versionskürzel (Attribut *Version.name*) im XML-Dokument im Headerbereich unter *header/document/software/specification@V* verwendet werden muss.

Nur eine Version ist für das Exportverfahren als gültig markiert. Die Version des Exportverfahrens kann eine Vorversion der gültigen Version der Spezifikation sein.

## 9.7 Administrative Objekte

Die Datenbank für QS-Dokumentation beinhaltet neben den Dokumentationsobjekten eine Gruppe von Tabellen, die Hilfestellungen für die Einhaltung eines korrekten Datenflusses und standardisierter Prüfprozesse geben.

Zu den administrativen Objekten gehören Mapping-Informationen mit denen QS-Daten in Form von Metainformationen zu administrativen Zwecken außerhalb des eigentlichen QS-Containers (dem Element *<qs-data>*) im XML verortet werden können (Abschnitt B 9.7.1). Außerdem gibt es



mit der Abfrage vPruefung eine Auflistung von Prüfungen, die für den jeweiligen Datenservice zu implementieren sind (Abschnitt B 9.7.3). Um eine Ansicht der administrativen Objekte zu erhalten, ist in Access (Spezifikationsdatenbank zur QS-Dokumentation) oberhalb der Objektübersicht das Drop-Down-Menü zu öffnen und der Menüpunkt „Benutzerdefiniert“ auszuwählen.

In einer separaten Datenbank ist eine Übersicht über die für ein Modul innerhalb einer Region relevanten technischen Datenservices mit Empfängeradressen und zu verwendenden XML-Schlüsseln enthalten (Abschnitt B 9.7.2).

### 9.7.1 XML-Mapping in der Spezifikationsdatenbank (QSDOK)

Exportfelder aus dem QS-Datensatz sind nicht immer Kind-Element von `case/qs_data` in der XML-Exportdatei. Die beiden Tabellen `ExportZiele` und `ExportZielXml` enthalten die Informationen, welche Exportfelder wohin verschoben, kopiert oder gelöscht werden. Die Tabelle `ExportFormatExportModul` zeigt neben diesen referenzierten Exportfeldern auch die Felder, die Kind-Element von `qs_data` bleiben.

Die Abfrage `vExportZieleXml` fasst die beiden Tabellen `ExportZiele` und `ExportZielXml` zu einer Übersicht zusammen. Hierbei ist zu beachten, dass nicht jedem Eintrag in der Tabelle `ExportZiele` auch ein Eintrag in der Tabelle `ExportZielXML` zugewiesen sein muss. Soll beispielsweise ein Datum aus dem Element `<qs_data>` lediglich gelöscht werden, ohne jedoch an andere Stelle im XML verschoben zu werden, ist keine dementsprechende `xmlXPath`-Angabe erforderlich.

Es können alle Feldarten referenziert sein. Referenziert sind aber nur solche Felder, die nicht oder nicht nur im Element `<qs_data>` aufgeführt werden. Die Felder werden referenziert und über diese Tabelle mit zusätzlichen Informationen in Bezug auf das XML verknüpft.

Die über das Attribut `fkFeld` referenzierten Felder gelten für alle Module, in denen dieses Feld verwendet wird. Wird ein Feld (gleichzeitig) über unterschiedliche Feldarten referenziert, überlagern detailliertere Angaben die allgemeinen. Wird so beispielsweise in der Tabelle `ExportZiele` ein Feld allgemein über den Fremdschlüssel `fkFeld` und in einer weiteren Zeile modulspezifisch über `fkBogenFeld` referenziert, werden die allgemeinen Regelungen (`loeschenQS`, `exportWennLeer`, `xmlXPath` und `xmlGruppierung`) für das Feld mittels der Referenzierung über `fkFeld` mit den spezifischeren Regelungen für das modulbezogene Feld mittels der Referenzierung über `fkBogenFeld` für das konkret aus der Referenzierung hervorgehende Modul überschrieben.

Dabei ist ebenfalls eine etwaige Konkretisierung über `fkExportModul` zu beachten, welche Ihrerseits konkretisierenden Charakter hat. Die über das Attribut `fkErsatzFeld` referenzierten Ersatzfelder gelten für alle Exportmodule, in denen dieses Ersatzfeld verwendet wird. Bei Referenzierung des gleichen Feldes (insbesondere bezogen auf `fkFeld`, `fkErsatzFeld` bzw. `fkZusatzfeld`) in zwei Zeilen – einmal ohne und einmal mit Angabe von `fkExportmodul` –

so überschreibt der Eintrag mit Angabe den Eintrag ohne Angabe für das konkret benannte Exportmodul.

Die booleschen Spalten `loeschenQS` und `exportWennLeer` geben folgende Informationen an:

- `loeschenQS`: Das Datum wird nur an den/die alternativen Ort(e) geschrieben und taucht in den QS-Daten nicht mehr auf. Hier geht es zum Beispiel um vom Datenschutz betroffene Felder, die nur in bestimmte Bereiche des XML geschrieben werden dürfen, wo sie dann verschlüsselt werden können.
- `exportWennLeer`: Hier wird ein leeres Feld nur dann berücksichtigt und leer eingetragen, wenn dieser Wert auf `true` steht.

In der Spalte `xmlXPath` vom Typ String steht ein XPath-Ausdruck, der ein Mapping auf einen im XML liegenden Ort des betroffenen Feldes enthält. Ausgangspunkt des XPath-Ausdrucks ist immer der aktuelle Datensatz im XML, also das `<case>` Element.

Die boolesche Spalte `xmlGruppierung` gibt an, dass ein neuer XML-Knoten nur eingesetzt wird, wenn das Datum einen neuen Wert enthalten sollte.

## 9.7.2 Datenservices

In einer separaten Datenbank sind Angaben über die beim Export relevanten technischen Datenservices und ihre verfahrensbezogene und regionale Zuordnung zu finden. Diese Datenbank zu Datenserviceinformationen ist hier zu finden:

<https://iqtig.org/datenerfassung/spezifikationen/spezifikation-zu-datenserviceinformationen/>

Die einzelnen Datenservices werden zusammen mit den E-Mail-Adressen, an die die Exportdateien zu versenden sind, und den für die Verschlüsselung der QS-Daten zu verwendenden XML-Schlüsseln in der Tabelle `Datenservice` abgebildet:

fkInstitution	Funktion	Sektor	Datenabsender	Email	Schlüssel
Geqik	DAS	Krankenhaus	LE	daten@geqik.de	<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_BW_LQS.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_BW_LQS.pub</a>
BAQ	DAS	Krankenhaus	LE	daten@baq-bayern.de	<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_BA_LQS.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_BA_LQS.pub</a>
GQH	DAS	Krankenhaus	LE	xmldaten@gqhnet.de	<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_HE_LQS.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_HE_LQS.pub</a>
PGS-QS	DAS	Krankenhaus	LE	qs-daten@nkgev.de	<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_NI_LQS.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_NI_LQS.pub</a>
SLAEK	DAS	Krankenhaus	LE	qs-daten@slaek.de	<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_SN_LQS.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_SN_LQS.pub</a>
KGSH	DAS	Krankenhaus	LE	proqs@kgsh.de	<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_SH_LQS.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_SH_LQS.pub</a>
BQS-Institut	DAS	Krankenhaus	LE	xmldaten@bqs-institut.de	<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_DL_BQS.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Datenannahmesstelle_DL_BQS.pub</a>
AQUA-Institut	BAS		VST		<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Bundesauswertungsstelle.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Bundesauswertungsstelle.pub</a>
SCI	VST		DAS		<a href="https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Vertrauensstelle.pub">https://www.sqg.de/downloads/2013/xml/schlüssel/Pub_key_Vertrauensstelle.pub</a>

Abbildung 8: Tabelle `Datenservice`

Die einzelnen Datenannahmestellen, die ihre Datenannahme mithilfe eines oder mehrerer dieser Datenservices realisieren, sind in der Tabelle `Region` hinterlegt, während die Zuordnung eines Datenservice zu einer Region abhängig vom Modul in der Tabelle `DatenserviceModul` festgelegt ist.

Eine zusammenfassende Darstellung der wichtigsten Merkmale ergeben sich aus den beiden Abfragen `DatenserviceExportverfahren` und `DatenserviceExportModule`. Die erste Abfrage aggregiert auf Ebene der Exportverfahren, während die

DatenserviceExportModule eine weitergehende Differenzierung für die einzelnen Exportmodule anbietet.

### 9.7.3 Prüfschritte

In der Gruppe administrative Objekte der QSDOK-Datenbank befindet sich die Abfrage vPruefung. vPruefung zeigt für die Institutionsarten LVKK/EK und BAS jeweils die Prüfungen, die wichtig für die Beurteilung der Güte der angelieferten Daten sind.

Die Datei datenflussprotokoll.xml in der Spezifikationskomponente XML-Schemata zeigt exemplarisch die Protokollierung der Prüfungen aus vPruefung. In der XML-Dokumentation ist die Protokollierung unter dem Xpath `root/header/protocol` zu finden.<sup>39</sup>

Tabelle 37: Felder der Abfrage vPruefung

Feldname	Bedeutung	Bezugselement im XML
Zielgruppe	Institutionsart, die die Prüfung ausführt	–
idPruefung	Referenz auf die konkrete einzelne Prüfung	–
pruefung	Kurzbeschreibung der Prüfung	optional unter: validation_provider/validation_item/@description
dpp	True/False, Prüfung ist im Datenprüfprogramm enthalten (11.1)	–
meta_DS	meta_DS soll ausdrücken, ob das Prüfergebnis das Persistieren der Metadaten (case/case_admin) eines Datensatzes verhindert. Wenn eine verletzte Prüfung die Eigenschaft meta_DS = true hat, dann werden die Metadaten nicht persistiert. Sind nur Prüfungen mit meta_DS = false verletzt, müssen die Metadaten aus case/case_admin unabhängig von status_document und status_case persistiert werden.	
beschreibung	ggf. erweiterte Erläuterungen zur Prüfung	–

<sup>39</sup> In der XML-Dokumentation sind die Kind-Elemente in der Zeile Model verlinkt.

Feldname	Bedeutung	Bezugselement im XML
fkFehlermeldung	Fehlerreferenz	validation_item/ error/rule_id/@V
Bereich	Einordnung der Prüfung	validation_ item/@V
strenge	Konsequenz im Fehlerfall (Hinweis, Fehler)	validation_item/ error/ rule_type/@V  In der Exportdatei ist ERROR auf H abzubilden und WARNING auf W.
regelverletzung	Bedingungen, die zu einer Re- gelverletzung führen	-
fehlermeldung	Standardisierte Fehlermel- dung	validation_item/ error/error_ message/@V
parameter	Parameter für Fehlermeldung	-
protokoll-Level	Zuordnung der Prüfung: DK-> betrifft Dokument DS-> betrifft Datensatz	-
datenfluss	direkt, indirekt, pid, sv, vpb	
verursacher	Verursacher des Fehlers	error/@originator
sortierung	mögliche Prüfungsreihen- folge	

## 10 XML-Schema

Die XML-Schema-Datei (XSD) ist eine Empfehlung des W3C<sup>40</sup> zum Definieren von Strukturen für XML-Dokumente.

Die Spezifikationskomponente XML-Schema enthält die XML-Schemata für die Schemavalidierung. Jede gelieferte Exportdatei wird von einer Datenannahmestelle gegen ein Schema validiert. Ein Datensatz - Element `case` und dessen Kind-Element - ist eine direkte Abbildung der Spezifikationskomponente QSDOK. Die Namen der XML-Elemente, ihre Reihenfolge, Häufigkeit, der Typ und der Elternknoten ergeben sich direkt aus der QSDOK.

Eine ausführliche Dokumentation der XML-Struktur liegt dem Ordner `documentation` der Spezifikationskomponente XML-Schema bei.

Im Ordner XML-Beispiele wird für jedes Exportmodul eine valide Exportdatei zur Verfügung gestellt.



### Hinweis

Pro Exportdatei ein Datensatz: Im Rahmen der Strukturabfrage darf ein Leistungserbringer nur einen Vorgang pro Jahr liefern. Die XML-Schemata zur Strukturabfrage fordern daher genau einen Datensatz pro Exportdatei. Aktualisierungen eines Vorganges sind unbegrenzt möglich.

---

---

<sup>40</sup> <http://www.w3.org/XML/Schema>

## 11 Tools

Das vorliegende Kapitel beschreibt Spezifikationskomponenten, die als Hilfsprogramme Prozesse in der Qualitätssicherung unterstützen. Die Hilfsprogramme basieren auf der Programmiersprache Java. Dementsprechend ist ein Abschnitt enthalten, der die Installation einer Java-Laufzeit-Umgebung (JRE) beschreibt. Die Tools selbst umfassen derzeit ein Verschlüsselungsmodul und ein Datenprüfprogramm.

### Java-Installation

Zur Ausführung von Java-Programmen ist die Java-Laufzeitumgebung (JRE) notwendig. Das JRE kann „online“ und „offline“ installiert werden.

Offizielle Installationspakete können von der Website <http://www.java.com/de/download/manual.jsp> heruntergeladen werden.

Anleitungen zur Installation von Java unter verschiedenen Betriebssystemen sind auf der folgenden Website detailliert beschrieben:

[https://www.java.com/de/download/help/download\\_options.xml](https://www.java.com/de/download/help/download_options.xml)

### Aufruf

Der Aufruf des Java-Programms erfolgt über die Konsole des Betriebssystems. Die Rückmeldungen des Programms erfolgen ebenfalls über die Konsole und können dort abgefangen werden.

Die Ein- und Ausgabe der Konsole lässt sich in der Regel in jedes Programm integrieren. Ein Exitcode von 0 ohne Ausgabe entspricht einer fehlerfreien Verarbeitung des Programms. Bei Fehlern gibt es einen Exitcode von 1 und in der Regel eine Konsolen- bzw. eine Error-Ausgabe.

Aufruf: `java -jar Beispielprogramm.jar -g -o Dateiname 2> error.txt`

- Java: Aufruf der JVM mit dem Befehl `java`
- `-jar`: Parameter `-jar`, mit dem der JVM mitgeteilt wird, dass ein Java-Archiv aufgerufen wird
- `Beispielprogramm.jar`: Benennung des Archivs, mit vollem Pfad, wenn es nicht im aktuellen Verzeichnis liegt
- `-g -o`: Parameter, die an das Java Programm übergeben werden sollen
- `2>`: Mit `2>` Ziel der Error-Ausgabe spezifizieren
- `error.txt`: Dateiname der Error-Ausgabe – hier die Datei „error.txt“

Bei längeren Pfaden oder Dateinamen, die ggf. Leerzeichen oder andere Zeichen enthalten, sind diese in Anführungszeichen zu setzen. Dies gilt sowohl für Paketnamen als auch für Parameter-Dateien.

## 11.1 Datenprüfprogramm

Das Datenprüfprogramm wird vor dem Hintergrund der Anforderungen der Richtlinien QFR, PPP und QSFFx bereitgestellt. Das Datenprüfprogramm bezieht die Plausibilitätsregeln direkt aus der Spezifikation.

Das Ausführen vom Datenprüfprogramm setzt Java in der Version 8 oder höher voraus.

### 11.1.1 Umfang der Prüfungen

Es werden zwei wesentliche Bereiche mit diesem Programm geprüft:

- Schema-Konformität (Struktur)
  - Die XML-Datei wird dabei auf Konformität mit dem zu Grunde liegenden Schema (XSD) überprüft.
- Regel-Konformität (Inhalte)
  - Die XML-Datei wird dabei auf Einhaltung der Regeln (XSLT) überprüft.

Die eigentliche Prüfung erfolgt in der XML-Export-Datei. Deren Struktur ist der entsprechenden Dokumentation bzw. dem gültigen XML-Schema zu entnehmen.

Die inhaltliche Prüfung selbst erfolgt über ein XSLT-Stylesheet und einen XSLT-Prozessor. Das Datenprüfprogramm verwendet die freie Version eines XSLT-Version 2.0-kompatiblen Programms (XSLT2). Die Einbindung von XSLT-Stylesheet und XSLT-Prozessor erfolgt über ein Java-Programm. Prinzipiell kann jeder XSLT2-fähige XSLT-Prozessor für die Prüfung auf dieser Grundlage verwendet werden. Das Datenprüfprogramm stellt eine Referenzimplementierung dar.

### 11.1.2 Programmaufruf

Das Datenprüfprogramm erzeugt eine Ausgabe/Output-Datei, die der Eingabe/Input-Datei entspricht, die jedoch um die Ergebnisse der Prüfungen erweitert wird. Die durchgeführten Prüfungen entsprechen einer Prüfung auf Dokumenten- und Vorgangsebene (Datensatzebene).

Das Datenprüfprogramm kann mehrere Dateien in einem Aufruf prüfen. Daher gibt es entsprechende Ordner für die Ein- und Ausgabedateien. Sollten diese Ordner nach der letzten Prüfung nicht geleert worden sein, so werden die Dateien des Eingabeordners erneut geprüft und der Ausgabeordner wird parallel mit Datum und Uhrzeit gesichert und ein neues leeres Ausgabeverzeichnis angelegt.

Die Prüfungen umfassen die Schemaprüfung und die Überprüfung der Feldinhalte (auch feldübergreifend).

### Parameter **-c** oder **-config**

Die Steuerung der Funktionen erfolgt über eine Konfigurationsdatei, deren Dateipfad dem Programm beim Programmstart mit dem Parameter **-c** oder **--config** beim Programmaufruf übergeben werden kann.

```
java -jar datenpruefprogramm-4.3.2-jar-with-dependencies.jar -c C:/konfiguration/config.xml
```

Wenn keine Konfigurationsdatei übergeben wird, wird die Datei `./config.xml` gesucht und geladen. Wenn diese Datei nicht gefunden wird, wird eine `Standard-config.xml`-Datei im Start-Order angelegt.

### Parameter **--no-spez-val**

Mit diesem Parameter wird das Prüfskript (XSL Stylesheet) ausgeschaltet.

```
java -jar datenpruefprogramm-4.3.2-jar-with-dependencies.jar -no-spez-val
```

### Parameter **--no-schema-val**

Mit diesem Parameter wird die Schemaprüfung ausgeschaltet.

```
java -jar datenpruefprogramm-4.3.2-jar-with-dependencies.jar -no-schema-val
```

### Batch-Dateien

Beim Datenprüfprogramm werden beispielhafte Batchdateien mitgeliefert:

- `datenpruefprogramm.bat`

Hier wird eine Datenprüfung anhand der Konfigurationsdatei „`config.xml`“ durchgeführt.

Die Konfigurationsdatei besteht aus den folgenden Bereichen:

### Provider (Softwareanbieter)

Im Element `<provider>` werden Daten benötigt, aus denen hervorgeht, wer das Prüfmodul einbindet und ausführt. Die Elemente `<fax>`, `<phone>`, und `<address>` sind optional, die anderen sind Pflichtelemente.

### GUI (Konsole)

Für ein vereinfachtes Debugging gibt es die Möglichkeit, eine Konsole mit detaillierten Programmausgaben während der Verarbeitung über das Element `<gui>` und den Wert `true` zu öffnen. Der Standard-Wert ist `false`.



### **Input\_Path (Eingabeverzeichnis) – überschreibbar mit Parameter -input**

Im Element `<input_path>` kann der Eingabeordner für die zu überprüfenden Exportdateien festgelegt werden. Das Element ist optional. Ohne diesen Parameter ist der Ordner `<arbeitsverzeichnis>\input\` der Standard-Eingabe-Ordner. Es werden alle Dateien mit der Dateiendung `.xml` verarbeitet. Wenn das Attribut `recursive` auf `true` steht, werden auch alle entsprechenden Dateien in Unterordnern berücksichtigt. Die Standard-Einstellung von `recursive` ist `false`.

### **Output\_Path (Ausgabeverzeichnis) – überschreibbar mit Parameter -output**

Im Element `<output_path>` kann der Ausgabeordner festgelegt werden. Das Element ist optional. Ohne diesen Parameter ist der Standard-Ausgabeordner `<arbeitsverzeichnis>\output\`. Der Dateiname der Ausgabedatei ist dabei gleich dem der Eingabedatei. Ein ggf. nicht vorhandener Ordner wird angelegt.

### **XSD\_Path (Schemaordner) – überschreibbar mit Parameter --xsd-path**

Im Element `<xsd_path>` wird der Schemapfad gesetzt.

### **XSL\_Path (XSLT-Stylesheet-Ordner) – überschreibbar mit Parameter --xsl-path**

Im Element `<xsl_path>` kann der Quellordner für die XSLT-Stylesheets festgelegt werden. Das Element ist optional. Ohne diesen Parameter wird im Standard XSL-Ordner `<arbeitsverzeichnis>\xsl\` nach den XSLT-Stylesheets gesucht.

Mehrere Konfigurationsdateien können für dasselbe Datenprüfprogramm angelegt werden, um beispielsweise Dateien unterschiedlicher Spezifikationen zu validieren oder die Durchführung einer Eingangs- bzw. einer Ausgangskontrolle jeweils vor der Entschlüsselung und nach der Verschlüsselung zu ermöglichen.

#### **11.1.3 Verzeichnisstruktur**

Für das korrekte Funktionieren des Datenprüfprogramms ist neben den erforderlichen Dateien auch eine korrekte Verzeichnisstruktur notwendig.

In der `config.xml` wird der `<xsl_path>` definiert.

Wenn der Parameter `<xsl_path>` auf ein Verzeichnis zeigt, muss in diesem Verzeichnis eine Stylesheet-Datei der folgenden Art vorliegen:

```
xsl\<Spezifikationsversion>.wxml
```

Hierbei handelt es sich um ein kompiliertes Haupt-XSL-Stylesheet, das die Prüfung entsprechend der Spezifikation durchführt. Es enthält alle Tests auf Regeln und Wertebereichsverletzungen. Ansonsten kann der Parameter `<xsl_path>` auf eine beliebige Stylesheet-Datei verweisen.

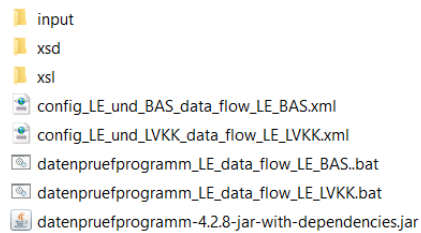


Abbildung 9: Beispiel einer typischen Verzeichnisstruktur

In Abbildung 9 ist eine typische Verzeichnisstruktur mit der config-Datei und dem Programm abgebildet. Unterhalb dieser Ordner befinden sich die Verzeichnisse, die schon in der Beschreibung der Konfiguration angesprochen wurden.

#### 11.1.4 Ausgabe

Nach dem Prüflauf des Datenprüfprogramms wird ein Ordner `<output>` erzeugt, der die geprüften Dateien und deren Datenflussprotokolle beinhaltet.

##### Geprüfte Dateien

Im Ordner `<output>/files` liegen nun die geprüften Quell-Dateien, welche um das Ergebnis der Prüfung erweitert worden sind. Jeder Datensatz wird innerhalb der XML-Datei geprüft und in der XML-Struktur abgelegt. Zudem wird ein neuer Eintrag als `validation_provider` erzeugt.

##### Protokolle

Im Ordner `<output>/protocol` liegen die Datenflussprotokolle. Sie entsprechen den geprüften Dateien im Ordner `<output>/files`, allerdings ohne QS-Daten und der Modifikation des provider-Elements.

##### HTML-Protokolle

Im Ordner `<output>/html` liegt eine `index.html`, in der auf vereinfachte Sichten der im Ordner `<output>/protocol` erstellten Protokollen verwiesen wird. Die HTML-Protokolle werden nicht an den Leistungserbringer versendet.

#### 11.1.5 Grafische Oberfläche

Wird der Parameter `GUI` in der Konfigurationsdatei auf „true“ gesetzt, wird das Datenprüfprogramm mit einer einfachen grafischen Oberfläche gestartet.

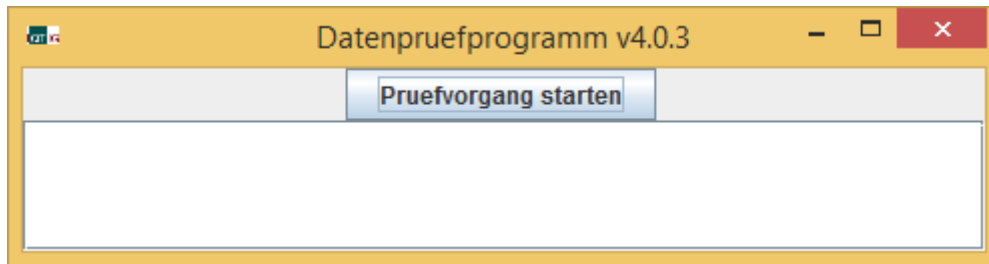


Abbildung 10: Grafische Oberfläche des Datenprüfprogramms

Bei der grafischen Oberfläche muss zum Starten die Schaltfläche „Pruefvorgang starten“ gedrückt werden.

Die grafische Oberfläche zeigt die Ausgabe im Fensterbereich direkt an. Am Inhalt des Ausgabe-Ordnerns ändert sich nichts; beide Laufvarianten (grafische Ausgabe oder Konsolenausgabe) erzeugen den gleichen Output.

## C Anhang

### Glossar

Begriff	Beschreibung
Bundesdatenpool	Zusammenführung aller bundesweit dokumentierten QS-Daten.
Config-Datei	Konfigurationsdatei, <u>Textdatei</u> in der bestimmte Einstellungen ( <u>Konfigurationen</u> ) gespeichert sind.
Datenannahmestelle	Stelle, an die die Leistungserbringer die erhobenen Daten übermitteln. Sie prüft die übermittelten Daten auf Plausibilität, Vollständigkeit und Vollzähligkeit.
Datenbasis	Im Einzelfall festzulegende bzw. festgelegte Gesamtmenge von auszuwertenden bzw. ausgewerteten Daten.
Datenfeld	Kleinste Einheit eines Datensatzes.
Datenfluss	Übermittlung der Daten der QS-Verfahren in einem festgelegten Format und Inhalt, die vom Leistungserbringer zur Datenannahmestelle gelangen. Die Datenflüsse sind grundsätzlich in der QFR-RL, PPP-RL und QSFFx-RL des G-BA festgelegt.
Datensatz	Datensatz entspricht einem case-Element aus einer Exportdatei. Ein Vorgang kann adressiert werden über die Vorgangsnummer (GUID). Ein Datensatz kann mit der Vorgangsnummer (GUID) - case/case_admin/guid/@V - und case/case_admin/version/@V adressiert werden.
Dokumentation	siehe: QS-Dokumentation
Einrichtung	siehe: Leistungserbringer
Gemeinsamer Bundesausschuss (G-BA)	Oberstes Beschlussgremium der gemeinsamen Selbstverwaltung der Ärzte, Zahnärzte, Psychotherapeuten, Krankenhäuser und Krankenkassen in Deutschland. Der G-BA bestimmt in Form von Richtlinien den Leistungskatalog der Gesetzlichen Krankenversicherung (GKV) für etwa 70 Millionen Versicherte und legt damit fest, welche Leistungen der medizinischen Versorgung von der GKV erstattet werden.
Leistungserbringer	Personen und Einrichtungen, die medizinische Versorgungsleistungen erbringen bzw. bereitstellen. Der Begriff wird im SGB V auch für Ärzte und ärztliche Einrichtungen, sowie für zugelassene Krankenhäuser gem. § 108 SGB V genutzt.
Leistungserbringeridentifizierende Daten (LID)	Daten, die eindeutig einen bestimmten Leistungserbringer identifizieren (z. B. Institutionskennzeichen oder Standortkennzeichen).
Plausibilitätsprüfung	Statistisches Verfahren, mit dem die Dokumentationsdaten auf erlaubte und/oder fehlende Werte, Widerspruchsfreiheit, Werteverteilung und bekannte Korrelationen geprüft werden.

Pseudocode	Programmcodex, der das zugrundeliegende Prinzip eines Algorithmus beschreibt, selbst aber nicht lauffähig ist. Er dient zur Veranschaulichung, unabhängig von der konkret zu verwendenden Programmiersprache.
QS-Daten	Sammelbegriff für alle Daten, die im Zuge eines QS-Verfahrens, z. B. einer Strukturabfrage nach QFR-RL, PPP-RL und/oder QSFFx-RL, erhoben und ausgewertet werden.
QS-Dokumentation	Gesonderte Erhebungen der QS-Daten durch die Leistungserbringer für die Qualitätssicherung.
Qualität	Bezogen auf die Gesundheitsversorgung: Grad, in dem versorgungsrelevante Ergebnisse, Prozesse und Strukturen bestimmte, definierte Anforderungen erfüllen.
Qualitätssicherung	Sammelbegriff für unterschiedliche Ansätze und Maßnahmen zur Sicherstellung festgelegter Qualitätsanforderungen bzw. zur Erreichung bestimmter Qualitätsziele. Hier: Gesetzliche Qualitätssicherung im Gesundheitswesen nach §§ 135-139 SGB V.
Qualitätssicherung, externe stationäre	Einrichtungsübergreifende Qualitätssicherung für medizinisch-pflegerische Leistungen, die ausschließlich im stationären Sektor erbracht werden.
Qualitätssicherungsmaßnahmen	Strukturierte, in Richtlinien geregelte Vorgehensweise, die Leistungserbringer bei der kontinuierlichen Qualitätsverbesserung unterstützt. Auslöser der Qualitätssicherungsmaßnahmen sind rechnerische Auffälligkeiten im Ergebnis eines Qualitätsindikators.
Regelbetrieb	auch: Routinebetrieb oder Echtbetrieb. Verpflichtende und flächendeckende Umsetzung eines QS-Verfahrens.
Sektor	Institutionell, d.h. durch unterschiedliche Abrechnungsmodalitäten getrennte Bereiche der medizinisch-therapeutischen Versorgung im deutschen Gesundheitswesen (z. B. ambulant/stationär).
Spezifikation	Datensatzbeschreibung. Festlegung, welche Daten für die Qualitätssicherung erhoben bzw. übermittelt werden müssen, welche Prüfalgorithmen zur Anwendung kommen (z. B. für Plausibilitätsprüfungen) und wie die Auslösung operationalisiert ist. Im Rahmen der Neuentwicklung von QS-Verfahren ist die Spezifikation als das Ergebnis der informationstechnischen Aufbereitung zu betrachten.
Systempflege	Routinemäßige und kontinuierliche Evaluation und Anpassung der Softwarespezifikation usw.
XSLT	Extensible Stylesheet Language Transformations. Programmiersprache zur Transformation von XML-Dokumenten in andere XML-Dokumente oder andere Dokumentformate wie HTML. Im QS-Kontext kann es auch für Datenprüfung und Protokollerstellung verwendet werden.