

# **Technische Dokumentation zur Spezifikation für die hessische QS-Dokumentation für Leistungserbringer**

**Erfassungsjahr 2024**



**Stand: 06. September 2023**

# Impressum

**Herausgeber:**

LAGQH – Landesarbeitsgemeinschaft Qualitätssicherung Hessen

**Thema:**

Technische Dokumentation zur Spezifikation für die hessische QS-Dokumentation für Leistungserbringer. Erfassungsjahr 2024

**Grundlage:**

Dieses Dokument wurde auf Basis der “Technischen Dokumentation zur Spezifikation für QS-Filter und QS-Dokumentation für Leistungserbringer, Erfassungsjahr 2014” vom 25. November 2013 erstellt. Herausgeber des Dokuments war

AQUA – Institut für angewandte Qualitätsförderung und Forschung im Gesundheitswesen GmbH  
Maschmühlenweg 8-10

37073 Göttingen.

**Anschrift des Herausgebers:**

LAGQH – Landesarbeitsgemeinschaft Qualitätssicherung Hessen

Frankfurter Straße 10-14

65760 Eschborn

Telefon: 06196 204 85 66

E-Mail: [info@lagqh.de](mailto:info@lagqh.de)

**Hinweis:**

Aus Gründen der leichteren Lesbarkeit wird im Folgenden auf eine geschlechtsspezifische Differenzierung verzichtet. Entsprechende Begriffe gelten im Sinne der Gleichbehandlung für beide Geschlechter.

# Inhalt

|  |    |
|--|----|
| Technische Dokumentation zur Spezifikation für die hessische QS-Dokumentation für Leistungserbringer ..... | 1  |
| 1. Einleitung.....   | 5  |
| 1.1. Zielsetzung der Technischen Dokumentation .....   | 5  |
| 1.2. Spezifikationsbegriff .....   | 5  |
| 1.2.1. Benennungsschema für Spezifikationspakete .....   | 5  |
| 1.2.2. Benennungsschema für Spezifikationskomponenten .....  | 6  |
| 2. QS-Filter .....   | 6  |
| 3. QS-Dokumentation.....   | 7  |
| 3.1. Zielsetzung der Spezifikation für die QS-Dokumentation.....   | 7  |
| 3.2. Anmerkungen zur Struktur der Spezifikation .....  | 8  |
| 3.2.1. Abfragen der Datenbank .....  | 8  |
| 3.2.2. Tabellenstruktur der Datenbank .....  | 9  |
| 3.3. Einrichtungsidifizierende Daten .....   | 10 |
| 3.4. Datenfeldbeschreibung.....  | 10 |
| 3.4.1. Module (Datensätze) .....   | 10 |
| 3.4.2. Bögen (Teildatensätze) .....  | 11 |
| 3.4.3. Datenfelder (Bogenfelder) .....   | 14 |
| 3.4.4. Felder – ein erster Schritt zur Prozess- und Datenintegration .....                                 | 15 |
| 3.4.5. Basistypen .....  | 16 |
| 3.4.6. Schlüssel .....   | 17 |
| 3.4.7. Schlüsselwerte.....   | 18 |
| 3.4.8. Überschriften .....   | 18 |
| 3.4.9. Ausfüllhinweise .....   | 19 |
| 3.4.10. Verwendung der Datenfeldbeschreibung für die Gestaltung von Eingabemasken.....                     | 19 |
| 3.5. Plausibilitätsprüfungen .....   | 21 |
| 3.5.1. Arten der Plausibilitätsprüfungen.....  | 21 |
| 3.5.2. Feldbezogene Prüfungen .....  | 22 |
| 3.5.3. Feldübergreifende Regeln .....  | 25 |
| 3.5.4. Regelsyntax .....   | 26 |
| 3.5.5. Teildatensatzübergreifende Regeln.....  | 31 |
| 3.5.6. Verfahren für die Evaluation von Regeln.....  | 31 |
| 3.5.7. Feldgruppen.....  | 32 |
| 3.5.8. Leitlinien für die Gestaltung der Benutzeroberflächen von Erfassungsprogrammen ....                 | 37 |
| 3.6. Listen von Schlüsselkodes (OPS, ICD-10-GM).....   | 38 |
| 3.6.1. OPS-Listen .....  | 39 |
| 3.6.2. ICD-Listen .....  | 39 |
| 3.7. Versionierung.....  | 40 |
| 3.7.1. Grundlegende Definitionen .....   | 40 |
| 3.7.2. Delta-Informationen zur vorhergehenden Version .....  | 40 |
| 3.7.3. Abgrenzung zwischen Erfassungsjahren und Datensatzformaten.....                                     | 42 |
| 3.7.4. Version des Exportverfahrens .....  | 42 |

|        |  |    |
|--------|--|----|
| 3.7.5. | Version der Exportdateien .....  | 43 |
| 4.     | Export .....   | 44 |
| 4.1.   | Datenexport .....  | 44 |
| 4.1.1. | Registrierung eines Dokumentationssystems .....                        | 44 |
| 4.1.2. | Identifizierung von Datensätzen .....                                  | 44 |
| 4.1.3. | Der Exportvorgang.....   | 45 |
| 4.1.4. | Export von Teildatensätzen .....                                       | 47 |
| 4.1.5. | Regeln für die Entgegennahme von Datensätzen und Teildatensätzen ..... | 50 |
| 4.1.6. | Die Antwortdateien .....   | 52 |

# 1. Einleitung

Die Technische Dokumentation zur Spezifikation für die QS-Dokumentation hessischer Leistungsbereiche richtet sich an die Leistungserbringer sowie an die mit der Umsetzung der Spezifikation beauftragten Softwarehersteller.

## 1.1. Zielsetzung der Technischen Dokumentation

Die Spezifikation für die QS-Dokumentation wird jeweils als Datenbank zur Erstellung der Software zur Verfügung gestellt. Das vorliegende Dokument erläutert die Struktur der Datenbanken und gibt Hilfestellung bei der Umsetzung.

## 1.2. Spezifikationsbegriff

Die Spezifikation ist die Gesamtheit aller Vorgaben, nach denen die QS-Dokumentation und die Übermittlung erfolgen soll, bezogen auf ein Erfassungsjahr.

Um die komplexen Anforderungen an die stationäre QS-Dokumentation sowie die zugehörigen Datenflüsse zu erfüllen, besteht die Spezifikation aus verschiedenen Komponenten, die je nach Anwender spezifisch zusammengestellt werden. Als Komponenten werden dabei Access-Datenbank, 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.

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 im nächsten Abschnitt 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 bei.

### 1.2.1. Benennungsschema für Spezifikationspakete

Die Benennung setzt sich wie folgt zusammen:

`<Erfassungsjahr>_<Richtlinie>_<Name>_<Betriebsart>_<Exportformat>_V<Versionsnummer>.zip`

Das Erfassungsjahr gilt für alle Spezifikationspakete und -komponenten, die Daten dieses Erfassungsjahres betreffen, egal in welchem Jahr das jeweilige Paket veröffentlicht wurde. Für die hessische QS-Dokumentation steht zurzeit im Platzhalter <Richtlinie> daher immer HE für Hessen-Spezifikation und für Name immer FDOK für fallbezogene QS-Dokumentation. Betriebsart ist immer RB – Regelbetrieb. Die Versionierung erfolgt in ganzen Zahlen, die zweistellig angegeben sind (unter 10 mit einer vorgestellten 0, z.B. V01, V02).

So wird beispielsweise im Herbst 2023 folgendes Paket veröffentlicht:

`2024_HE_FDOK_RB_CSV_V01.zip`

Ausformuliert bezeichnet dies die Spezifikation der hessischen, fallbasierten QS-Dokumentation für das Erfassungsjahr 2024 für den Regelbetrieb. Als Exportart ist CSV vorgesehen. Es handelt sich um die erste Version. Um ein solches Paket beispielsweise im Gespräch kurz zu bezeichnen, kann man sagen: die Spezifikation 2024 V01.

### 1.2.2. Benennungsschema für Spezifikationskomponenten

Die Benennung der Spezifikationskomponenten lehnt sich an das bei den Spezifikationspaketen verwendete Prinzip an.

<Erfassungsjahr>\_<Art der Komponente>\_V<Versionsnummer>.<Dateierweiterung>

Die Art der Komponente bezieht sich auf die jeweilige Funktion und wird durch ein Kürzel angegeben.

Folgende Bezeichnungen stehen für die Art der Komponenten:

- **TechDok\_LE** – bezeichnet die technische Dokumentation für Leistungserbringer; diese gibt detaillierte Erläuterungen zur Funktionsweise und Verwendung der einzelnen Komponenten. Da es verschiedene spezifisch auf eine jeweilige Zielgruppe hin verfasste technische Dokumentationen gibt, wird die Zielgruppe gleich im Kürzel vermerkt: das LE bedeutet "Leistungserbringer".
- **HE\_QSDOK** – die Access-Datenbank, in der die hessische QS-Dokumentation spezifiziert wird.
- **Ausfüllhinweise** – auf der Komponentenebene ist dies eine zip-Datei, welche die Versionierung und vollständige Bezeichnung enthält. Sie enthält einzelne HTML-Dateien für jeden Leistungsbereich, die mit den Kürzeln der einzelnen Leistungsbereiche benannt sind.
- **Dokuboegen** – auf der Komponentenebene ist dies eine zip-Datei, die die Versionierung und vollständige Bezeichnung enthält. Sie enthält die Dokumentationsbögen als einzelne PDF-Dateien für jeden Leistungsbereich, die mit den Kürzeln der einzelnen Leistungsbereiche benannt sind.

Die Versionierung erfolgt in ganzen Zahlen, die zweistellig angegeben sind (unter 10 mit einer vorgestellten 0, z.B. V01, V02).

## 2. QS-Filter

Die QS-Filter-Spezifikation der hessischen Leistungsbereiche ist in die bundesweite QS-Filter-Spezifikation des IQTIG integriert. Weitere Informationen erhalten Sie über folgende URL:

<https://iqtig.org/datenerfassung/spezifikationen/qs-basispezifikation-fuer-leistungserbringer>.

In der QS-Filter-Spezifikation sind die QS-Filter für die hessischen Module SA\_HE, SA\_FRUEHREHA\_HE und MRE\_HE definiert.

## 3. QS-Dokumentation

### 3.1. Zielsetzung der Spezifikation für die QS-Dokumentation

Mit der vorliegenden Spezifikation für die Erstellung von Software zur Erfassung, Plausibilitätsprüfung und Übermittlung von Daten für die externe vergleichende Qualitätssicherung in Hessen soll die Bereitstellung valider und vergleichbarer Daten gewährleistet werden.

Externe Qualitätssicherungsmaßnahmen, die einen Vergleich der Qualität von Krankenhausleistungen zum Ziel haben, stellen eine Reihe von methodischen Anforderungen an die Datenerhebung, Datenerfassung und Plausibilitätsprüfung, um valide, zuverlässige und auch vergleichbare Daten gewinnen zu können. Die Erfassung und Plausibilitätsprüfung durch unterschiedliche Programme beinhalten grundsätzlich die Gefahr einer Verzerrung der Daten. Die Vorgaben dieser Spezifikation sollen dazu dienen, durch einheitliche Festlegung von Datenfeldbeschreibungen, Plausibilitätsregeln, Grundsätzen der Benutzerschnittstellengestaltung und Datenübermittlungsformate dieser Gefahr entgegenzuwirken.

Diese Dokumentation richtet sich an Hersteller und Entwickler von Krankenhaussoftware, interessierte Krankenhausmitarbeiter und an alle anderen Interessenten.

Hersteller von Krankenhaussoftware entwickeln Programme, welche die Erfassung aller qualitätsrelevanten Daten dieser Spezifikation ermöglichen, diese Daten auf Vollständigkeit und Plausibilität prüfen, im vorgegebenen Format an die LAGQH exportieren und fehlerhafte (von der LAGQH abgelehnte) Datensätze in geeigneter Weise dem Anwender zur Korrektur und erneutem Export vorlegen. Sowohl die LAGQH als auch die Hersteller von Krankenhaussoftware müssen die Daten auf Plausibilität und Vollständigkeit prüfen.

Darüber hinaus bietet die Spezifikation den Softwareanbietern Hinweise für die Gestaltung der Benutzeroberfläche und die benutzerfreundliche Plausibilitätsprüfung unmittelbar bei der Dateneingabe. Die Datenstellen nehmen die Qualitätsdaten von den Krankenhäusern entgegen und prüfen sie auf Vollständigkeit und Plausibilität. Nach erfolgreicher Prüfung werden Datensätze von den Datenstellen als angenommen bestätigt. Fehlerhafte Datensätze werden in einem Fehlerprotokoll angezeigt. Folgende Übersicht listet die Bedeutung einzelner Themen dieses Dokuments für die beiden Zielgruppen auf:

Bedeutung verschiedener Themen für die Zielgruppen der Spezifikation

| Thema          | relevant für Daten-<br>entgegennahme (LAGQH) | relevant für<br>Softwarehersteller |
|----------------|--|------------------------------------|
| Bogenfelder    | ja   | ja                                 |
| Felder         | weniger                                      | ja                                 |
| Basistypen     | ja   | ja                                 |
| Regeln         | ja   | ja                                 |
| Mehrfachregeln | ja   | weniger                            |
| Syntax         | ja   | ja                                 |
| RegelFelder    | ja   | ja                                 |
| Feldgruppen    | nein   | ja                                 |
| Exportformat   | ja   | ja                                 |

## 3.2. Anmerkungen zur Struktur der Spezifikation

Die QS-Dokumentation-Spezifikation ist in einer relationalen Datenbank abgelegt. Zurzeit wird sie ausschließlich als Access-Datenbank (MS Access 2000) zur Verfügung gestellt. Der Name der Spezifikation richtet sich nach folgendem Schema:

<Erfassungsjahr>\_HE\_QSDOK\_V<Versionsnummer>.mdb

<Erfassungsjahr> bezeichnet das Jahr, in dem die QS-Dokumentation stattfindet.  
<Versionsnummer> bezeichnet die 2-stellige Versionsnummer (z. B. 01, 02).

### Beispiel:

Im Erfassungsjahr 2024 ist die Spezifikation 2024\_HE\_QSDOK\_V01.mdb gültig.

Weitere Erläuterungen finden Sie in der Einleitung zum allgemeinen Teil: Benennungsschema für Spezifikationskomponenten (Abschnitt 1.2.2).

### 3.2.1. 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.
- Datenfeldbeschreibung: Hier sind alle Bogenfelder der spezifizierten Module, sortiert nach Modulname, Bogenname und Zeilennummer der Bogenfelder dargestellt (Abschnitt 3.4).
- DatenfeldbeschreibungFürEinModul: Wenn man diese Abfrage aufruft, muss der Modulname (z. B. „MRE\_HE“) 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 3.5).
- PlausibilitätsregelnFürEinModul: Wenn man diese Abfrage aufruft, so muss der Modulname (z. B. „SA\_HE“) 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 3.4.2).
- Ersatzfelder: Dies ist eine Auflistung der zu anonymisierenden Bogenfelder für alle spezifizierten Module (Abschnitt 4.1.4).
- OPSListen: Diese Abfrage liefert einen Überblick über die Codes der OPS-Listen (Abschnitt 3.6.1).
- ICDListen: Hier sind die Codes der ICD-Listen dargestellt (Abschnitt 3.6.2).
- Exportfelder: Wenn man diese Abfrage aufruft, erhält man eine Übersicht über alle Exportfelder (Abschnitt 4.1.3).
- ExportfelderFürEinModul: Diese Abfrage zeigt eine Auswahl der Exportfelder eines Moduls (Modulname ist explizit anzugeben). Man erhält eine Übersicht über die zu exportierenden Felder inkl. der Zuordnung zum Teildatensatz.
- Feldgruppen: Diese Abfrage liefert eine Übersicht über alle Feldgruppen (Abschnitt 3.5.7).
- FeldgruppenFürEinModul: Wenn man diese Abfrage aufruft, so muss der Modulname (z. B. „SA\_FRUEHREHA\_HE“) 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 direkt 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 direkt angegeben werden.

### 3.2.2. Tabellenstruktur der Datenbank

Die Tabellen und Spalten (Attribut) 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.

#### Beispiele:

- BasisTyp (Tabelle)
- idBasisTyp (Spalte)

Für jede Tabelle ist in der Spezifikation höchstens ein Primärschlüssel definiert, der nach folgendem Schema aufgebaut ist:

id<TabellenName>

Der Ausdruck in den eckigen 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, welches durch Setzen eines weiteren, eindeutigen Index (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 „Nachschlagetabellen“ 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 Tabellen erstrecken (Inklusionsverknüpfungen, Joins).

Die Fremdschlüsselattribute (Namen beginnen mit fk) wurden in MS Access als Datenbankattribute zum Nachschlagen eingerichtet. Beispielsweise wird beim Fremdschlüsselattribut fkModul in der Tabelle Bogen nicht mehr der Primärschlüssel des jeweiligen Moduls, sondern der Name des Moduls angezeigt. Diese Änderung betrifft nur die Anzeige, nicht die Struktur der Datenbank. Sind zwei Tabellen mehrfach durch Schlüssel-Fremdschlüssel-Beziehungen miteinander verknüpft, so kann der Name eines Fremdschlüssels auch folgendermaßen aufgebaut sein:

fk<FremdTabellenName><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, jedoch einen eindeutigen Schlüssel, der über die Fremdschlüsselfelder definiert ist. „Primärschlüssel“ wird hier im Sinne der Access-Definition eines Primärschlüssels benutzt. Streng genommen wird über die beiden

Fremdschlüssel ein neuer Primärschlüssel definiert. Ein Beispiel hierfür ist die Tabelle Regelfelder, welche die Tabellen BogenFeld und Regel verknüpft. Folgende Attribute treten in vielen Tabellen auf und seien hier kurz erläutert:

- name ist in der Regel als „technischer Name“ zu verstehen. Beispielsweise wird Feld.name als Variablenname in den Plausibilitätsregeln verwendet.
- bezeichnung ist eine kurze Beschreibung. Beispielsweise 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.

### 3.3. Einrichtungsidentifizierende Daten

In Hinblick auf eine einrichtungs- und standortbezogene Auswertung und Berichterstattung sind einrichtungsidentifizierende Daten in der QS-Dokumentation zu dokumentieren. Wichtigstes Merkmal dazu ist das Institutskennzeichen. Als Grundlage einer standortbezogenen Sichtweise gibt es das Feld STANDORT, in dem die StandortId des entlassenden Standorts anzugeben ist. Die Standortkennzeichnung des Falles muss in der QS-Dokumentation und der Sollstatistik übereinstimmen. Die entsprechende Organisation und Übernahme des Wertes in die QS-Dokumentation sind durch die Software sicherzustellen.

### 3.4. Datenfeldbeschreibung

Für alle Dokumentationsbögen eines Moduls existiert jeweils eine eigene Datenfeldbeschreibung. Sie spezifiziert alle auszufüllenden Datenfelder (Bogenfelder, auch Items genannt) und besteht aus mehreren Tabellen, welche in den nachfolgenden Abschnitten erläutert werden. Ein Dokumentationsbogen oder kurz Bogen ist als eine Menge von auszufüllenden Datenfeldern zu verstehen. Die Papierform ist hier nur als eine Erscheinungsform des Dokumentationsbogens zu verstehen. Man spricht besser von Eingabeformular oder Eingabemaske.

Die Abfragen Datenfeldbeschreibung und DatenfeldbeschreibungFürEinModul der Access-Datenbank ermöglichen den Überblick auf diese Struktur. Das für den Anwender wichtigste Merkmal ist die Bezeichnung des Datenfeldes (Attribut BogenFeld.bezeichnung).

Die Datenfeldbeschreibung orientiert sich eng am Dokumentationsbogen („Bogensicht“). Grundsätzlich ist die „Bogensicht“ die Sicht der medizinischen Fachgruppen, welche die Module entwickeln. Bei verteilten Softwarelösungen für das Krankenhaus ist die Bogensicht dann nicht mehr adäquat, wenn die Bestandteile eines Bogens auf verschiedene Teilsysteme verteilt sind. Die Daten eines Bogens werden für den Export aus den einzelnen Teilsystemen zusammengestellt. Die Papierbögen werden lediglich als Layoutinformation zur Verfügung gestellt. Sie sind zur Dokumentation nicht zugelassen.

Im Kontext einer integrierten, prozessorientierten Krankenhaussoftware 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 des Krankenhauses einordnet.

#### Ziele

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

#### 3.4.1. Module (Datensätze)

Ein Modul der Spezifikation enthält die Datensatzdefinition mindestens eines medizinischen Leistungsbereiches (Beispiele: Schlaganfall, Multiresistente Erreger). Die QS-

Dokumentationssoftware kann für einen Behandlungsfall eine Moduldokumentation anlegen, welche nach Dokumentationsabschluss an die LAGQH übermittelt wird. Fehlerfreie Moduldokumentationen (verkürzt „Module“) werden dem Krankenhaus von der LAGQH bestätigt. Die Landesauswertung basiert auf den Moduldokumentationen der Krankenhäuser.

Aus technischer Sicht ist das 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. Es werden nur die Felder erläutert, die in Hessen genutzt werden.

Struktur der Tabelle Modul:

| Feldname              | Feldtyp   | Bemerkung  |
|-----------------------|-----------|--|
| idModul               | INTEGER   | Primärschlüssel  |
| name                  | TEXT(32)  | Eindeutiger technischer Name   |
| bezeichnung           | TEXT(255) | Erläuternde Bezeichnung  |
| verpflichtend         | BOOLEAN   | Besteht für das Modul eine QS-Dokumentationsverpflichtung in Hessen?   |
| primaerModul          | BOOLEAN   | Ist das Modul ein Primärmodul? (in Hessen immer true)  |
| mehrfachDokumentation | BOOLEAN   | Ist ein mehrfaches Anlegen eines gleichartigen Datensatzes pro Krankenhausfall zulässig (ja/nein)? (in Hessen immer false) |
| ahtext                | MEMO      | Einleitender Text für den Ausfüllhinweis eines Moduls (in Hessen nicht verwendet)  |
| fkVersion             | INTEGER   | Gültige Version des jeweiligen Moduls  |
| Bild                  | TEXT(20)  | Modulspezifisches Bild (in Hessen nicht verwendet)   |

### Auslösung der Moduldokumentation

Der auslösende Sachverhalt für die Dokumentationspflicht ist in der Spezifikation für QS-Filter-Software definiert. Die QS-Filter-Software greift zu diesem Zweck auf administrative Routinedaten (z.B. Haupt- und Nebendiagnosen und Prozeduren) zurück, welche in jedem Krankenhausinformationssystem (KIS) verfügbar sind und von den Krankenhäusern auch für die Umsetzung der Datenübermittlungsvereinbarung gemäß §301 Abs. 3 SGB V (kurz: DÜV-301) benötigt werden.

### Primärmodule – Minimaldatensatz

Für die hessischen Module SA\_HE, SA\_FRUEHREHA\_HE und MRE\_HE dürfen ab der Spezifikation 2023 keine Minimaldatensätze mehr angelegt werden.

### 3.4.2. Bögen (Teildatensätze)

Die Begriffe Teildatensatz und Bogen werden als Synonyme gebraucht. Ein Teildatensatz entspricht bei der Papierdokumentation einem Teil eines Dokumentationsbogens. Teildatensätze können theoretisch mehrfach pro Datensatz ausgefüllt werden. Dies ist in Hessen jedoch nicht der Fall.

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:

Struktur der Tabelle Bogen

| Feldname | Feldtyp | Bemerkung |
|----------|---------|-----------|
|----------|---------|-----------|

|                      |         |   |
|----------------------|---------|---|
| idBogen              | INTEGER | Primärschlüssel   |
| name                 | TEXT    | Technischer Name des Teildatensatzes  |
| bezeichnung          | TEXT    | Beschreibender Text   |
| existenzBedingung    | MEMO    | Logische Bedingung (Regeln für das Anlegen von Teildatensätzen)   |
| fkModul              | INTEGER | Obligatorischer Fremdschlüssel zu einem Modul   |
| fkBogenZahl          | TEXT(1) | Anzahl der auszufüllenden Teildatensätze pro Patient (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.          |
| fkEindeutigBogenfeld | 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 Bogenamen identifiziert und angesprochen:

<Modul.name>:<Bogen.name>

#### Beispiele:

- MRE\_HE:B ist der Basisbogen des Moduls "Multiresistente Erreger"
- SA\_HE:HI\_TIA ist der Hirninfarkt-Teildatensatz des Moduls Schlaganfall Hessen (SA\_HE)

### Bogentyp

Der Kerndatensatz besteht aus mindestens einem Basisteildatensatz und kann durch einen oder mehrere Teildatensätze ergänzt werden. Das Attribut fkBogenTyp definiert für jeden Teildatensatz seine Rolle im und seine Zugehörigkeit zum Kerndatensatz.

Inhalte der Tabelle BogenTyp:

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

### Hierarchie von Teildatensätzen

Der Ausgangspunkt für die Teildatensatzhierarchie eines Moduls ist immer der Basisteildatensatz (Wert B des Attributs fkBogenTyp). Ein abhängiger Teildatensatz besitzt einen Mutterteildatensatz, der über das Attribut fkMutterBogen definiert ist. 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). 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.

Beispielsweise definiert die Ausprägung `fkBogenZahl = *` eine 1-n-Beziehung. Man beachte, dass das Attribut `fkBogenZahl` wichtig für das Verfahren der Entgegennahme von Datensätzen ist (Abschnitt 6.1.5).

#### Beispiele:

- Der Teildatensatz `MRE_HE:B` muss als Basisteildatensatz genau einmal ausgefüllt werden (`fkBogenZahl = 1`).
- Der Teildatensatz `SA_HE:HI_TIA` darf höchstens einmal pro Datensatz angelegt werden (`fkBogenZahl = ?`).

Man beachte weiterhin, dass die im Attribut `fkBogenZahl` definierten Kardinalitäten durch Definitionen in den nachfolgend beschriebenen Attributen `existenzBedingung` bzw. `fkEindeutigBogenFeld` eingeschränkt werden können.

### **Inhaltliche Voraussetzung für das Anlegen von Teildatensätzen**

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

Die Krankenhaussoftware muss die Existenzbedingung als Trigger für das Anlegen eines abhängigen Teildatensatzes nutzen. Wenn die Existenzbedingung eines potenziellen Kindteildatensatzes erfüllt ist, so muss der Kindteildatensatz auch angelegt und übermittelt werden.

Andererseits gilt: Wenn die LAGQH einen Kindteildatensatz erhält, für den die zugehörige Existenzbedingung im Mutterteildatensatz nicht erfüllt ist, so ist das eine relationale Plausibilitätsverletzung.

#### Beispiel: Modul SA\_HE

Der Teildatensatz `SA_HE:HI_TIA` darf nur innerhalb eines Vorgangs angelegt werden, wenn im zugehörigen Mutterteildatensatz `SA_HE:B` folgende Bedingung erfüllt ist:

```
FALLABSCHLUSS = LEER UND LEFT(KLAS_ICD;3) IN ('G45'; 'I63'; 'I64') UND
alter(GEBDATUM;AUFNDATUM) >= 18
```

Wenn also der Patient älter als 18 Jahre ist, die Diagnose mit G45, I63 oder I64 beginnt und das Feld `FALLABSCHLUSS` leer ist,

- muss der abhängige Teildatensatz `SA_HE:HI_TIA` angelegt werden.
- darf der abhängige Teildatensatz `SA_HE:ICB_SAB` nicht angelegt werden.
- muss ein bereits angelegter Teildatensatz `SA_HE:ICB_SAB` wieder gelöscht werden.

### **Identifizierende Attribute mehrfach vorhandener Teildatensätze**

Teildatensätze, welche 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, welches im Attribut `fkEindeutigBogenFeld` festgelegt wird. In Hessen wird dieses Feld derzeit nicht verwendet. Bei der entgegennehmenden Stelle kommt noch das Feld `RegistrierNr` hinzu, da dort Datensätze verschiedener Krankenhäuser gesammelt werden.

Beim Anlegen einer Tabelle für die Speicherung eines mehrfach vorhandenen Teildatensatzes muss

der Primärschlüssel mindestens die Attribute VorgangsNr, VersionNr und das in fkEindeutigBogenFeld definierte Feld umfassen.

### 3.4.3. 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 wird so gewählt, dass sie einem medizinischen Experten unmittelbar verständlich ist. Die Spezifikation des Inhalts umfasst dagegen sowohl eine fachliche (medizinische) als auch datentechnische Typisierung. Dagegen repräsentieren die in der Tabelle Feld aufgelisteten Felder inhaltlich gleiche Dokumentationsfelder mehrerer Module. Der datentechnische Typ (BasisTyp) charakterisiert das Format des Feldes.

Jedes Datenfeld hat zwingend einen Bezug zu einem Teildatensatz und zu einem medizinischen Feld. Der Name des Datenfeldes ist identisch mit dem Namen des medizinisch fachlichen Feldes. Weitere Eigenschaften sind der Text, welcher das Item auf dem Erhebungsformular kennzeichnet, und die fortlaufende Nummer im Teildatensatz. Die Datenfelder sind in der Tabelle BogenFeld gespeichert.

Identifizierendes Merkmal eines Datenfeldes 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. Das heißt, dass eine Abfrage mit dem Primärschlüsselpaar (modulNr, feldNr) höchstens einen Primärschlüssel idBogenFeld liefert.

Die Tabelle BogenFeld ist eine Verknüpfungstabelle, die wegen ihrer zentralen Bedeutung – im Gegensatz zu anderen Verknüpfungstabellen – eine Ausnahmeregelung besitzt: Diese äußert sich darin, dass hier ein Primärschlüssel idBogenFeld definiert ist.

Struktur der Tabelle BogenFeld:

| Feldname      | Feldtyp  | Bemerkung  |
|---------------|----------|--|
| idBogenFeld   | INTEGER  | Primärschlüssel  |
| fkBogen       | INTEGER  | Fremdschlüssel zu dem Teildatensatz und zu dem Feld, bilden zusammen die identifizierenden Merkmale  |
| fkFeld        | INTEGER  |  |
| zeileAufBogen | DOUBLE   | Zeile in dem Dokumentationsbogen   |
| bezeichnung   | TEXT     | Beschreibender Text zum Feld auf dem Dokumentationsbogen. Wenn der Feldinhalt leer ist, wird der Inhalt des gleichnamigen Feldes in der Tabelle Feld genommen. |
| elemente      | INTEGER  | Anzahl der Elemente bei Listenfeldern  |
| fkMussKann    | TEXT(1)  | „M“ oder „K“, Unterscheidung zwischen Muss- und Kann-Feldern   |
| min           | DOUBLE   | Harte Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulspezifisch). Die Definition ist optional.  |
| max           | DOUBLE   | Harte Obergrenze des Wertebereichs eines numerischen Datenfeldes (modulspezifisch). Die Definition ist optional.   |
| minWeich      | DOUBLE   | Weiche Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulspezifisch). Die Definition ist optional.   |
| maxWeich      | DOUBLE   | Weiche Obergrenze des Wertebereichs eines numerischen Datenfeldes (modulspezifisch). Die Definition ist optional.  |
| ahinweis      | TEXT(32) | Name des HTML-Ausfüllhinweises ohne Endung .htm (Abschnitt 4.5.3)  |

#### Muss- und Kann-Felder



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

- Ein Muss-Feld muss innerhalb eines angelegten Teildatensatzes immer ausgefüllt sein.
- Kann-Felder deklarieren dagegen abhängige Felder in Feldgruppen und müssen nur unter bestimmten Bedingungen ausgefüllt werden. Wenn also logische Sachverhalte dem Ausfüllen von Kann-Feldern entgegenstehen, so dürfen sie nicht ausgefüllt werden.

#### Anzahl der Elemente von Listenfeldern

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

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

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

Die Einführung der Tabelle `Feld` erleichtert dem Softwarehersteller den Abgleich seines Datenmodells mit dem Datenmodell der Spezifikation. Redundante Information auf der Menge aller Dokumentationsbögen muss dadurch nicht redundant abgebildet werden.

Beispielsweise taucht das Feld `ENTLGRUND` (Entlassungsgrund) in den meisten Modulen auf. Um die mehrfache Pflege dieser Felder zu vermeiden, wird ein Feld mit dem Namen `ENTLGRUND` definiert und jeweils nur noch in der Tabelle `BogenFeld` referenziert.

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` beschrieben. Das Attribut `bezeichnung` ist ein Standardtext für das gleichnamige Attribut der Tabelle `BogenFeld`. Im Eingabeformular wird die Bezeichnung aus der Tabelle `BogenFeld` angezeigt.

Identifizierendes Attribut eines Feldes ist allein sein technischer Name (Attribut `name`). Dieses ist ganz wichtig für die Eindeutigkeit von Feldnamen innerhalb eines Moduls. Felder mit unterschiedlichen Typen oder unterschiedlichen Schlüsseln müssen auch 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. Man beachte die Besonderheiten der Listenfelder beim Datenexport und in der Syntax der Plausibilitätsregeln. Die Anzahl der Elemente des von einem Feld abgeleiteten Bogenfeldes wird über das Attribut `elemente` der Tabelle `BogenFeld` gesteuert.

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

Struktur der Tabelle `Feld`:

| Feldname    | Feldtyp | Bemerkung  |
|-------------|---------|--|
| idFeld      | INTEGER | Primärschlüssel  |
| Name        | TEXT    | Technischer Name   |
| bezeichnung | TEXT    | (Erlaubte Zeichen: A-Z, 0-9, Ziffer nicht am Anfang)<br>Beschreibender Text auf dem Dokumentationsbogen<br>(Standardwert für gleichnamiges Feld in Tabelle<br>BogenFeld) |
| laenge      | INTEGER | Anzahl der Zeichen in der Feldeingabemaske, enthält beim<br>Typ <code>ZAHL</code> auch das Komma, bei <code>SCHLUESSEL</code> die<br>Trennzeichen                        |

|                 |          |  |
|-----------------|----------|--|
| nachKommaLaenge | INTEGER  | Anzahl der Nachkommastellen in der Feldeingabemaske (muss kleiner als laenge sein)                                   |
| fkBasisTyp      | INTEGER  | Fremdschlüssel zur Tabelle Basistypen  |
| fkSchluessel    | INTEGER  | Fremdschlüssel zur Tabelle Schlüsseltypen  |
| istListe        | BOOLEAN  | Wenn istListe wahr ist, so sind die vom betreffenden Feld abgeleiteten Bogenfelder Listenfelder.                     |
| fkKombiFeld     | INTEGER  | Optionalen Fremdschlüssel auf ein anderes Feld, welches Kombinationsfelder kennzeichnet                              |
| einheit         | TEXT(50) | Einheit des Feldes (z. B. mm, Stunden)   |
| formatAnweisung | TEXT     | Formatanweisung zum Ausfüllen des Datenfeldes (z. B. TT.MM.JJJJ)   |
| min             | DOUBLE   | Harte Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulübergreifend). Die Definition ist optional.  |
| max             | DOUBLE   | Harte Obergrenze des Wertebereichs eines numerischen Datenfeldes (modulübergreifend). Die Definition ist optional.   |
| minWeich        | DOUBLE   | Weiche Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulübergreifend). Die Definition ist optional. |
| maxWeich        | DOUBLE   | Weiche Obergrenze des Wertebereichs eines numerischen Datenfeldes (modulübergreifend). Die Definition ist optional.  |

### Kombinationsfelder

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

### 3.4.5. Basistypen

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

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

Struktur der Tabelle BasisTyp:

| Feldname           | Feldtyp | 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 Feld (einschließlich Vorzeichen und Komma) |
| stdNachKommaLaenge | INTEGER | Vorschlagsfeld für das gleichnamige Feld in der Tabelle Feld                                       |

Anmerkungen:



- In Zeichenketten (Basistyp TEXT) sind alle Zeichen des ASCII-Formats mit einem Kode > 32 erlaubt. Ausgenommen sind das Semikolon, die doppelten Anführungsstriche und Hochkommata.
- Es gibt zwei Arten von Schlüsseln: numerisch und nichtnumerisch (Abschnitt 3.4.6).
- Das Komma trennt die Nachkommastellen, Vorzeichen + und – sind erlaubt.

### 3.4.6. Schlüssel

Identifizierendes Merkmal eines Schlüssels (Kodesystem) ist sein technischer Name. Die meisten Schlüsselkodes sind in der Tabelle SchlüsselWert (Abschnitt 3.4.7) definiert.

Struktur der Tabelle Schlüssel:

| Feldname           | Feldtyp | Bemerkung   |
|--------------------|---------|---|
| idSchlüssel        | INTEGER | Primärschlüssel   |
| name               | TEXT    | Technischer Name (muss eindeutig sein)  |
| bezeichnung        | TEXT    | Beschreibender Text   |
| extern             | BOOLEAN | Zeigt an, ob der Schlüssel in der Tabelle Schlüssel (= falsch) oder in einer externen Tabelle gespeichert (= wahr) ist.       |
| externVerweis      | TEXT    | Verweis auf die Quelle eines externen Schlüssels  |
| zahl               | BOOLEAN | Wenn wahr, sind die Werte im Attribut code der zugehörigen Schlüsselwerte als ganze Zahl kodiert, ansonsten als Zeichenkette. |
| sortierNrVerwendet | BOOLEAN | Flag, welches anzeigt, ob für die Reihenfolge das Attribut sortierNr der Tabelle SchlüsselWert herangezogen wird.             |
| fkMutterSchlüssel  | INTEGER | Referenz auf einen übergeordneten Schlüssel. In Hessen nicht verwendet.   |

Schlüsselkodes 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 3.5.4).

#### Externe Schlüsselkataloge

Externe Schlüsselkataloge sind über das Attribut extern deklariert. Externe Schlüsselkataloge werden nicht von der LAGQH bereitgestellt und somit auch nicht verantwortet.

Achtung: Der Softwareanbieter und die LAGQH haben dafür Sorge zu tragen, dass die aktuellen externen Schlüsselkataloge in der Software verwendet werden.

Hinweise zu den Bezugsquellen sind in der Spalte externVerweis zu finden (z. B. <http://www.dimdi.de>). Ein Verweis auf eine Bezugsquelle kann unabhängig vom Attribut extern angegeben werden (siehe Schlüssel EntlGrund).

Der Fachabteilungsschlüssel (Fachabt) ist ein externer Schlüsselkatalog: Die Schlüsselcodes, welche der LAGQH zum Zeitpunkt der Publikation der QS-Spezifikation bekannt sind, sind in der Tabelle SchlüsselWert enthalten. Spätere Schlüsseländerungen bzw. Fortschreibungen müssen vom Softwareanbieter und von der LAGQH selbstständig und zeitnah über die §301-Vereinbarung (<http://www.dkgev.de>) bezogen werden.

Der Schlüssel Entlassungsgrund (EntlGrund) basiert auf einem externen Schlüssel, der als Schlüssel 5 in Anlage 2 der §301-Vereinbarung definiert ist: Die 1. und 2. Stelle dieses §301-Schlüssels werden im Rahmen der QS-Spezifikation numerisch kodiert (siehe Attribut zahl). Dabei sind nicht alle Schlüsselwerte des Schlüssels 5 in Anlage 2 der §301-Vereinbarung für die QS-Dokumentation zulässig.

Achtung: Der Schlüssel EntlGrund ist kein externer Schlüssel (extern = FALSE). Das bedeutet, dass die in der Spezifikation enthaltenen Werte in der Software zu verwenden sind, auch wenn diese von dem Entlassungsgrund nach §301 abweichen.

Die Schlüsselcodes sind in der Tabelle SchluessselWert enthalten. Spätere Schlüsseländerungen bzw. Schlüsselfortschreibungen werden von der LAGQH zeitnah übernommen.

### 3.4.7. Schlüsselwerte

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

Struktur der Tabelle SchluessselWert:

| Feldname          | Feldtyp  | Bemerkung   |
|-------------------|----------|---|
| idSchluessselWert | INTEGER  | Primärschlüssel   |
| fkSchluesssel     | INTEGER  | Fremdschlüssel zur Tabelle Schlüssel  |
| code              | TEXT(50) | Schlüsselcode (entweder numerisch oder alphanumerisch kodiert)  |
| bezeichnung       | TEXT     | Textliche Definition des Schlüsselwertes  |
| sortierNr         | INTEGER  | Optionale Angabe zur Reihenfolge der Schlüsselwerte: Wenn belegt, so ist diese Reihenfolge bei der Anzeige in der Erfassungssoftware einzuhalten. |

Das Attribut code ist ein Textfeld, welches in Abhängigkeit vom Wert des Attributes zahl im zugeordneten Schlüssel entweder numerisch oder nichtnumerisch interpretiert wird. Wenn in einer Plausibilitätsregel (Abschnitt 3.5) Felder mit numerischen Schlüsseln (Basistyp NUMSCHLUESSEL) vorkommen, so werden bei der Evaluierung der Regel die Schlüsselcodes wie ganze Zahlen behandelt. Beispielsweise ist der numerische Schlüsselcode 07 gleichwertig mit 7. Bei einem nichtnumerischen Schlüsselfeld (Basistyp SCHLUESSEL) hätten die beiden Codes eine unterschiedliche Bedeutung.

#### Sortierung der Codes

Für die Codes (Attribut SchluessselWert.code) eines Schlüssels ist eine Sortierung definiert. Die Art der Sortierung wird über die Attribute zahl und sortierNrVerwendet der Tabelle Schluesssel festgelegt.

- Numerische Sortierung: Wenn sortierNrVerwendet = nein und zahl = ja, so sind die Codes nach der Spalte code der Tabelle Schluesssel numerisch zu sortieren.
- Alphanumerische Sortierung: Wenn sortierNrVerwendet = nein und zahl = nein, so sind die Codes nach der Spalte code der Tabelle Schluesssel alphanumerisch zu sortieren.
- Spezielle Sortierung: Wenn sortierNrVerwendet = ja, so sind die Codes nach den Werten in der Spalte sortierNr der Tabelle Schluesssel numerisch zu sortieren.

### 3.4.8. Überschriften

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

Struktur der Tabelle Abschnitt:

| Feldname    | Feldtyp | Bemerkung       |
|-------------|---------|-----------------|
| idAbschnitt | INTEGER | Primärschlüssel |

|                  |         |   |
|------------------|---------|---|
| bezeichnung      | TEXT    | Text der Überschrift  |
| ebene            | INTEGER | Zeigt die Hierarchie der Überschriften an                         |
| fkStartBogenFeld | INTEGER | Fremdschlüssel auf das erste zur Überschrift gehörende Bogenfeld  |
| fkEndeBogenFeld  | 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.

Die in der Spezifikationsdatenbank hinterlegten Überschriften sind in die Eingabemasken der QS-Dokumentationssoftware zu integrieren.

#### Achtung:

Viele Datenfelder sind für den Anwender erst im Kontext der Überschriften verständlich.

### 3.4.9. Ausfüllhinweise

Die Ausfüllhinweise zu den Datenfeldern sind in einem separaten Zip-Archiv enthalten, welches nach dem Benennungsschema für Spezifikationskomponenten bezeichnet wird (Abschnitt 1.2.2). 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

#### Beispiel:

Das Bogenfeld 2504 (Spezifikation 2024) hat in der Spalte `ahinweis` den Eintrag „IDNRPAT“. Der zugeordnete Ausfüllhinweis des Zip-Archivs heißt `IDNRPAT.htm`.

Wenn der Eintrag in `ahinweis` leer ist, so existiert für das betreffende Bogenfeld kein Ausfüllhinweis.

Das Attribut `fkAhinweisTyp` lässt die Differenzierung von drei verschiedenen Arten von Ausfüllhinweisen zu:

- Feldbezogen: Der Ausfüllhinweis bezieht sich auf den entsprechenden Datensatz in der Tabelle `Feld`. Der Ausfüllhinweis ist modulunabhängig (Beispiel: `IDNRPAT.htm`)
- Modulspezifisch: Soll sich ein Ausfüllhinweis nur auf ein bestimmtes Modul beziehen, kann der Attributwert modulspezifisch ausgewählt werden (Beispiel: `ENTLGRUND$MRE_HE.htm`). Der Ausfüllhinweis bezieht sich nur auf das Modul `MRE_HE`.
- Speziell: Soll es für verschiedene Datenfelder der Tabelle `Feld` einen gemeinsamen Ausfüllhinweis geben, kann dieser als speziell deklariert werden. Der Attributwert `ahinweis` definiert den Namen des Ausfüllhinweises.

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.

### 3.4.10. Verwendung der Datenfeldbeschreibung für die Gestaltung von Eingabemasken

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

Diese Spezifikation definiert als Minimalstandard die für den Anwender sichtbaren Inhalte der

Dokumentationsformulare. Als Referenz für die sichtbaren Inhalte dienen die Dokumentationsbögen, welche zusammen mit der Spezifikation von der LAGQH veröffentlicht werden. Die Dokumentationsbögen werden als PDF-Dokumente bereitgestellt, welche aus der Spezifikationsdatenbank automatisch generiert worden sind.

Die folgende Tabelle 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.

Informationen aus der Datenbank, welche im GUI verwendet werden:

| <b>Tabelle</b> | <b>Attribut</b>         | <b>Bemerkung</b>  | <b>sichtbar für Anwender</b> |
|----------------|-------------------------|---|------------------------------|
| Modul          | name                    | Kürzel des Datensatzes (z. B. SA_HE), erscheint üblicherweise im Titel des Formulars.   | ja                           |
| Modul          | bezeichnung             | Bezeichnung des Datensatzes (z. B. Schlaganfall-Akutbehandlung), erscheint üblicherweise im Titel des Formulars.  | ja                           |
| Bogen          | bezeichnung             | Bezeichnung des Teildatensatzes   | ja                           |
| BogenFeld      | zeileAufBogen           | Nummer des Datenfeldes. Dient bei umfangreicheren Bögen zur besseren Orientierung.  | (ja)                         |
| BogenFeld      | bezeichnung             | Bezeichnung des Datenfeldes   | ja                           |
| BogenFeld      | ergaenzende Bezeichnung | Ergänzende Bezeichnung zum Datenfeld, kann z.B. durch Wahl der Schrift von der Bezeichnung abgesetzt werden   | ja                           |
| Feld           | laenge                  | Definiert die Länge des Eingabefeldes. Für die Gestaltung des Eingabefeldes sind weitere Informationen aus der Datenbank wichtig (z. B. Feld.nachKommaLaenge).  |                              |
| Feld           | einheit                 | Einheiten (wie z. B. ml) müssen angezeigt werden.   | ja                           |
| Feld           | format Anweisung        | Formatanweisungen (wie z. B. TT.MM.JJJJ) sollen – sofern nicht durch übergeeignete Eingabefelder unterstützt – für den Anwender angezeigt werden.   | (ja)                         |
| SchluesselWert | 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)                         |
| SchluesselWert | bezeichnung             | Bei Schlüsselfeldern müssen die Textdefinitionen der Codes (z. B. in einer Auswahlliste) angezeigt werden. Für die Sortierung sind die Attribute sortierNrVerwendet und zahl der Tabelle Schluessel relevant (Abschnitt 3.4.6). | ja                           |
| Abschnitt      | bezeichnung             | Die Überschriften sind wichtig für die Strukturierung und das Verständnis des Datensatzes und müssen deshalb in der QS-Dokumentationssoftware angezeigt werden.   | ja                           |

Werden Datenfelder (z. B. OPS) eines QS-Datensatzes aus Fremdsystemen über Schnittstellen importiert, so sollen die übernommenen Daten auch in der Erfassungsssoftware 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.

### 3.5. Plausibilitätsprüfungen

Fehlende und widersprüchliche Angaben in den Datensätzen sollen durch umfangreiche Plausibilitätsprüfungen verhindert werden. In der QS-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 – Übermitteln der Datensätze an die LAGQH, dortige Prüfung, ggf. Übermitteln eines Fehlerberichts an das dokumentierende Krankenhaus, Korrektur der Dokumentation und erneutes Übermitteln des Datensatzes – weitgehend entfällt.

Die LAGQH führt für jeden Datensatz alle harten Plausibilitätsprüfungen der Spezifikation durch. Bei einer Regelverletzung ist der Datensatz zurückzuweisen.

Es gelten folgende Grundsätze der Plausibilitätsprüfung:

- Alle Felder müssen ausgefüllt sein, wenn andere logische Sachverhalte dem nicht 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; Fehlermeldungen werden vorgegeben).
- Keine Angabe (bzw. kein Feldinhalt) wird ergänzt oder gelöscht.

#### 3.5.1. Arten der Plausibilitätsprüfungen

Es wird zwischen drei Arten von Plausibilitätsprüfungen unterschieden:

- harte Prüfungen
- weiche Prüfungen in der QS-Dokumentationssoftware
- warnende Prüfungen bei der Datenentgegennahme

Die drei Arten der Prüfungen werden in unterschiedlichen Kontexten (QS-Software bzw. Datenentgegennahme) durchgeführt und haben unterschiedliche Konsequenzen. Die folgende Tabelle gibt einen Überblick:

Arten der Plausibilitätsprüfungen

| Art der Prüfung | Kürzel | Prüfung durch QS-Software | Prüfung durch Datenstelle | Konsequenz: verhindert Dokumentationsabschluss oder Datenentgegennahme |
|-----------------|--------|---------------------------|---------------------------|--|
| hart            | H      | ja                        | ja                        | ja   |
| weich           | W      | ja                        | optional                  | nein   |
| warnend         | D      | optional                  | ja                        | nein   |

#### Harte Prüfungen

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

- ein Dokumentationsabschluss der QS-Dokumentation eines Vorgangs unzulässig oder
- 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 oder warnend gekennzeichnet sind.

### **Weiche Plausibilitätsprüfungen**

Die weichen Plausibilitätsprüfungen der Spezifikation sind von der QS-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.

### **Warnende Plausibilitätsprüfungen**

Unterjährig (in weiteren Versionen während des Erfassungsjahres) findet keine Verschärfung der harten Plausibilitätsprüfungen statt, um den Anwendern in den Krankenhäusern den Dokumentationsabschluss nach transparenten Regeln zu ermöglichen.

Trotz einer Vielzahl harter Plausibilitätsprüfungen kann es aber dennoch vorkommen, dass Krankenhäuser inhaltlich unplausible Datensätze liefern. Fallen derartige Lücken in der Datenqualität nach Start des Erfassungsjahres auf, so kann die im Krankenhaus installierte QS-Dokumentationssoftware den Anwender nicht auf diesen Mangel hinweisen.

Warnende Plausibilitätsprüfungen können von der LAGQH unterjährig in weiteren Versionen publiziert werden, um die Krankenhäuser auf gravierende Mängel in der Datenqualität hinzuweisen. Warnende Plausibilitätsprüfungen sind somit als ernste Warnungen zu verstehen.

Folgende Aufgaben sind von den beteiligten Stellen umzusetzen:

- Die LAGQH setzt die warnenden Plausibilitätsprüfungen um.
- Krankenhäuser erhalten über das Fehlerprotokoll Informationen über die mangelhaften Datensätze.
- Die QS-Dokumentationssoftware zeigt die Warnungen den Anwendern nach der Rückprotokollierung an.
- Anwender haben dann die Möglichkeit, die mit Warnungen versehenen Datensätze zu korrigieren und erneut einzusenden.

### **3.5.2. Feldbezogene Prüfungen**

Feldbezogene Prüfungen sind Plausibilitätsregeln, welche sich auf ein einziges Datenfeld beziehen.

Einige feldbezogene Prüfungen – insbesondere die Wertebereichsüberprüfungen – können auch in der formalen Regelsyntax formuliert werden (Beispiel: SCNIHSSA > 25), welche erst in nachfolgenden Abschnitten behandelt wird.

Die in diesem Abschnitt behandelten Prüfungen ergeben sich direkt aus der Datenfeldbeschreibung (Tabellen Feld und BogenFeld).

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

Bei der Datenentgegennahme sind alle harten feldbezogenen Prüfungen zu evaluieren und bei Regelverletzung die unten definierten standardisierten Fehlertexte im Rahmen des Korrekturverfahrens an das Krankenhaus zu übermitteln.

Die nachfolgend beschriebenen feldbezogenen Prüfungen werden vor Evaluation der in Abschnitt 3.5.3 beschriebenen feldübergreifenden Regeln durchgeführt.

### **Überprüfung des Formats**

Die Formatprüfung bezieht sich auf den Exportdatensatz: Die QS-Dokumentationssoftware muss Daten im korrekten Format generieren, die LAGQH prüft auf Korrektheit des Formats.

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:

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<BogenFeld.name> "<BogenFeld.bezeichnung>" (Zeile <BogenFeld.zeileAufBogen>) ist kein gültiger <Basistyp.name> Wert (<Basistyp.bezeichnung> <Basistyp.format>).

Beispiel:

Der Wert '3A.06.2012' des Datenfeldes SA\_HE:B:AUFNDATUM "Aufnahmedatum Krankenhaus" (Zeile 9) ist kein gültiger DATUM-Wert (Zehnstelliges Datum TT.MM.JJJJ).

### Überprüfung der Feldlänge

Überschreitet die Anzahl der Zeichen eines Wertes die spezifizierte Länge (Attribut laenge in Tabelle Feld) des Feldes?

Standardisierter Fehlertext für Längenfehler:

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

Beispiel:

Der Wert '31.06.20040' des Datenfeldes SA\_HE:B:AUFNDATUM "Aufnahmedatum Krankenhaus" (Zeile 9) überschreitet die zulässige Feldlänge 10.

### Überprüfung der Schlüsselcodes

Werden bei Schlüsselfeldern nur zulässige Schlüsselcodes verwendet?

Standardisierter Fehlertext bei unzulässigen Schlüsselcodes:

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<BogenFeld.name> "<BogenFeld.bezeichnung>" (Zeile <BogenFeld.zeileAufBogen>) ist kein zulässiger Kode des Schlüssels <Schlüssel.name> (<Schlüssel.bezeichnung>).

Beispiel:

Der Wert 'xyz' des Datenfeldes MRE\_HE:B:ENTLDIAG „Entlassungsdiagnosen“ (Zeile 34) ist kein zulässiger Kode des Schlüssels ICD10 (ICD-10-GM SGB V).

Besonderheiten bei externen Schlüsseln:

- Bei externen DIMDI-Schlüsseln (ICD-10-GM oder OPS) sind die jeweils gültigen amtlichen Kataloge zu verwenden. Alle Codes eines Behandlungsfalls müssen in derjenigen Katalogversion dokumentiert sein, welche am Aufnahmetag Krankenhaus des Patienten gültig ist.
- Nicht-terminale ICD- oder OPS-Kodes sind unzulässig!
- Fehlen bei OPS-Kodes Seitenlokalisationen, obwohl diese erforderlich sind, so ist der OPS-Kode fehlerhaft.

### 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. Falls auch in der Tabelle BogenFeld Grenzen definiert sind (= not null), werden die in der Tabelle BogenFeld gesetzten speziellen Wertgrenzen statt der Grenzen in der Tabelle Feld angewandt.



Standardisierter Fehlertext bei Unterschreitung einer Wertebereichsgrenze:

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

Beispiel:

Der Wert '1' des Datenfeldes SA\_HE:ICB\_SAB:SCGCS\_A\_BL "Glasgow Coma Scale Summenscore" (Zeile 91) ist kleiner als '3'.

Standardisierter Fehlertext bei Überschreitung einer Wertebereichsgrenze:

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

Beispiel:

Der Wert '45' des Datenfeldes SA\_HE:HI\_TIA:SCNIHSSA "NIH-Stroke Scale bei Aufnahme" (Zeile 30) ist größer als '42'.

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

### Überprüfung der Muss-Felder des Datensatzes

Ein nicht ausgefülltes Muss-Feld (Kapitel 3.4.3) führt zu einer Regelverletzung.

Standardisierter Fehlertext für Muss-Fehler:

Das Datenfeld <Modul.name>:<Bogen.name>:<BogenFeld.name> "<BogenFeld.bezeichnung>" (Zeile <BogenFeld.zeileAufBogen>) muss einen gültigen Wert enthalten.

Beispiel:

Das Datenfeld SA\_HE:B:AUFNDATUM "Aufnahmedatum Krankenhaus" (Zeile 9) muss einen gültigen Wert enthalten.

Achtung:

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

### 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, welche nicht pseudonymisiert werden und keine Listenfelder 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 (FELD\_1) ein Muss-Feld, die weiteren Elementfelder (FELD\_2, FELD\_3 etc.) sind Kann-Felder.
- Zusatzfelder der Tabelle ExportFormat sind Muss-Felder, außer wenn sie Schlüsselfelder mit einem Jleer-Schlüssel (z. B. das Zusatzfeld Storno) sind.

Das redundante Feld fkMussKann in der Tabelle ExportFormat zeigt an, ob das entsprechende Feld ausgefüllt sein muss.

Beispiel:

Das Feld entlquartal berechnet sich über die Formel quartal(ENTLDATUM). In Modulen, in denen das Datenfeld ENTLDATUM ein Muss-Feld ist, ist auch das Ersatzfeld ein Muss-Feld und die LAGQH ist



verpflichtet, die Muss-Prüfung hier auch durchzuführen. Ansonsten ist das Ersatzfeld ein Kann-Feld.

Für Exportfelder (Tabelle ExportFormat), welche einen Bezug zu einem Ersatzfeld (Tabelle ErsatzFeld) bzw. zu einem Zusatzfeld (Tabelle ZusatzFeld) haben, sind die standardisierten Fehlertexte anzupassen:

Standardisierter Fehlertext für Muss-Fehler eines Ersatzfeldes:

Das Ersatzfeld des Datenfeldes <Modul.name>:<Bogen.name>:<BogenFeld.name> "<BogenFeld.bezeichnung>" (Zeile <BogenFeld.zeileAufBogen>) 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.

### 3.5.3. Feldübergreifende Regeln

Die 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. Eine Plausibilitätsregel müsste eigentlich „Unplausibilitätsregel“ heißen, weil sie unplausible Zustände beschreibt, die zu Fehlermeldungen führen.

Um die Regeln von den später definierten Feldgruppenregeln (Abschnitt 3.5.7) zu unterscheiden, werden sie auch Einzelregeln genannt.

#### Die Regeltabelle

Die Bedingungen für unplausible Angaben sind in der Tabelle Regeln abgelegt. Die Syntax ist in Abschnitt 3.5.4 beschrieben. Die Bedingungen sind möglichst kurzgefasst (Vermeidung von durch ODER verknüpften Teilbedingungen). Jede Bedingung kommt nur einmal innerhalb eines Moduls vor.

Struktur der Tabelle Regeln:

| Feldname          | Feldtyp | Bemerkung  |
|-------------------|---------|--|
| idRegeln          | INTEGER | Primärschlüssel  |
| fkModul           | INTEGER | Fremdschlüssel zur Tabelle Modul   |
| bedingung         | MEMO    | Entsprechend der Syntax definierte Regeln  |
| meldung           | MEMO    | Fehlermeldung: Diese Texte sind bei Regeln mit Bezug zu Feldgruppen automatisch generiert.   |
| alternativMeldung | MEMO    | Alternative Fehlermeldung: Wenn hier ein Text vorhanden ist, so ist dieser anstelle des Textes in der Spalte meldung zu verwenden. |
| fkMehrfachRegel   | INTEGER | Fremdschlüssel zur Tabelle mit mehrfach vorkommenden Regeln  |
| fkFeldGruppe      | INTEGER | Optionalen Fremdschlüssel zur Tabelle FeldGruppe: Indikator dafür, ob eine Regel aus einer Feldgruppe generiert wurde.             |

|                   |         |   |
|-------------------|---------|---|
| fkRegelTyp        | TEXT(1) | Fremdschlüssel zur Tabelle RegelTyp:<br>Die Regeltypen sind die in Abschnitt 3.5.1 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D   |
| gueltigNachExport | BIT     | Regeln, welche den Wert nein haben, können von der LAGQH nicht evaluiert werden. Stattdessen werden die referenzierten Ersatzbedingungen der Tabelle MehrfachRegel evaluiert (falls definiert). |

### Bogenfelder einer Regel

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

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

Struktur der Tabelle RegelFelder

| Feldname    | Feldtyp | Bemerkung   |
|-------------|---------|---|
| fkBogenFeld | INTEGER | Fremdschlüssel zu den Tabellen Feld und Regeln, bilden zusammen den Primärschlüssel |
| fkRegeln    | INTEGER |   |

### Mehrfach vorkommende Regeln

Wenn in einer Regel von der Pseudonymisierung betroffene Datenfelder benutzt werden, so können diese von der LAGQH nicht evaluiert werden. Stattdessen wird für solche Regeln aus der Tabelle Regeln über einen Bezug zur Tabelle MehrfachRegel eine Ersatzbedingung definiert, welche von der LAGQH zu evaluieren ist.

Struktur der Tabelle MehrfachRegel:

| Feldname        | Feldtyp | Bemerkung   |
|-----------------|---------|---|
| idMehrfachRegel | INTEGER | Primärschlüssel   |
| bedingung       | MEMO    | Entsprechend der Syntax definierte Regeln   |
| meldung         | MEMO    | Kontextunabhängige Fehlermeldung  |
| ersatzBedingung | MEMO    | Ersatzregel für den pseudonymisierten Datensatz   |
| fkRegelTyp      | TEXT(1) | Fremdschlüssel zur Tabelle RegelTyp:<br>Die Regeltypen sind die in Abschnitt 3.5.1 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D |

### Beispiel:

Die Datenfelder der Regel 9 (AUFNDATUM > ENTLDATUM) werden nicht zwingend an die LAGQH übermittelt. Bei der Entgegennahme ist die in der Tabelle MehrfachRegel definierte Ersatzregel (idMehrfachRegel = 87) anzuwenden: vwDauer < 0.

### Weitere Regeln

Weitere feldübergreifende Regeln sind die in Abschnitt 3.4.2 beschriebenen Existenzbedingungen für das Anlegen von abhängigen Teildatensätzen (Attribut existenzBedingung in Tabelle Bogen).

### 3.5.4. Regelsyntax

Bedingungen sind in den Tabellen Regeln, MehrfachRegel und Bogen definiert. Die den Bedingungen zugrunde liegende Regelsyntax wird in diesem Abschnitt beschrieben. Jede Regel ist ein logischer

Ausdruck, welcher das Ergebnis WAHR oder FALSCH hat. Jede Regel bezieht sich auf einen angegebenen 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.

## Typen

Die möglichen Typen der Datenfelder sind in der folgenden Tabelle aufgelistet.

Datentypen der Datenfelder in den Plausibilitätsregeln

| Datentyp      | Bezeichnung   | Beispiele (Literals) |
|---------------|---|----------------------|
| BOOL          | Boolsche Variable:                                  | WAHR, FALSCH         |
| TEXT          | Zeichenkette (String)                               | "Spezifikation"      |
| GANZEZAHL     | ... -2, -1, 0, 1, 2, 3, ... (auch negative Zahlen!) | 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'               |
| UHRZEIT       | Uhrzeit   | '10:15'              |

In der Spezifikation für 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 dieser nicht in Hochkommata gesetzt werden.
- Schlüsselwerte, die alphanumerische Codes beinhalten, haben den Basistyp SCHLUESSEL. Beispielsweise die OPS-Schlüssel (z. B. '5-282.0') oder 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.

## 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 enthalten sein. Bei den Ersatzregeln in Tabelle Mehrfachregel sind stattdessen die Exportfelder des Moduls erlaubt. Bei der Evaluierung von Regeln werden die aktuellen Werte der referenzierten Bogenfelder eingesetzt. Kann-Bogenfelder können auch nicht ausgefüllt 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 positiv ist. Ansonsten 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.

Ein Beispiel ist das Feld ENTLDIAG, welches im Modul MRE\_HE als Bogenfeld vorkommt. Hinter ENTLDIAG verbirgt sich eine Liste mit 25 Elementen, welche nachfolgend als Variable in einer Regel angesprochen wird:

ENTLDIAG KEINSIN MRE\_ICD

Andererseits können auch die Elemente einer Liste einzeln angesprochen werden:

ENTLDIAG\_1, ENTLDIAG\_2, ..., ENTLDIAG\_n

Die Nummer des jeweiligen Elements (zwischen 1 und n) wird mit vorhergehendem Unterstrich an den Schluss des Feldnamens gehängt. Jedes Element kann z.B. innerhalb einer Plausibilitätsregel einzeln angesprochen werden.

Die einzelnen Elemente sind als Exportfelder in der Tabelle ExportFormat enthalten.

### Literale

Alphanumerische Literale (z.B. SCHLUESSEL) werden von einfachen Hochkommata eingeschlossen, während Zeichenketten vom Datentyp TEXT in doppelte Anführungsstriche gesetzt werden müssen. Beim Export entfallen die begrenzenden Zeichen!

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

#### Beispiele für Regeln:

GEBDATUM <= '01.01.1890'

CTMRANGIO NICHTIN (1;2) UND DOPPLERDUPLEX NICHTIN (1;2)  
UND GEFVERSCHL <> LEER

### Listen von Literalen

Literale können sowohl als Skalare als auch als Listen angesprochen werden. Der Separator einer Liste von Literalen ist das Semikolon.

#### Beispiele:

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

Um zu prüfen, ob alle Listenfelder ausgefüllt sind, wird die Liste über den Wert LEER angesprochen.

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 (z.B. MRE\_ICD) und die darin enthaltenen Prozedur- und Diagnosecodes beinhalten (Abschnitt 3.6).

#### Beispiel:

In der Regel 2089 aus Modul SA\_HE (Schlaganfall-Akutbehandlung) wird die ICD-Liste SA\_ICD verwendet.

KLAS\_ICD NICHTIN SA\_ICD

Außerdem gibt es Teildatensatz-Listenfelder, welche im Abschnitt 3.5.5 beschrieben werden.

### Operatoren

Die folgende Tabelle gibt einen groben Überblick über die in der Syntax zulässigen Operatoren. Der aktuelle Überblick über alle zulässigen Operationen (inkl. Operanden) ist in Tabelle SyntaxOperator der Spezifikation zu finden.

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 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     |  |

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

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

#### Anmerkungen:

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

nur rechts: IN, NICHTIN

links und rechts: EINSIN, KEINSIN, JEDESIN, EINSNICHTIN

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.

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).

#### Beispiele:

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

#### Hinweis

Folgende Operatoren sind komplementär:

- IN und NICHTIN
- EINSIN und KEINSIN
- JEDESIN und EINSNICHTIN

#### Beispiel:

Folgende Ausdrücke sind gleich:

- A EINSNICHTIN B
- NICHT A JEDESIN B

### **Funktionen**

Eine Funktion ist gekennzeichnet durch ihren Namen, an den sich unmittelbar (ohne Leerzeichen) ein Listenausdruck anschließt. Funktionen ohne Übergabeparameter werden ähnlich wie in C oder Java durch ein Klammerpaar abgeschlossen.

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

Häufig wird nur die Signatur von Funktionen bereitgestellt. Weitere Hinweise für die Implementierung findet man in Kapitel 5 (Anhang A). 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, welche 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)) vor.

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

Die OPS- und ICD-Listen enthalten ausschließlich Normcodes. Die im Krankenhaus dokumentierten Codes enthalten ggf. auch Zusatzkennzeichen (Bsp. Seitenlokalisierung). Bei der Evaluation der Regeln werden die dokumentierten Zusatzkennzeichen ignoriert (Abschnitt 3.6).

#### Beispiel:

Die OPS-Liste KAT OPS enthält unter anderem den Code 5-144.01. Die Evaluation der Regel OPSCHLUESSEL EINSIN KAT OPS führt auch dann zu einem positiven Ergebnis, wenn OPSCHLUESSEL = ('5-144.01:R';LEER;LEER;LEER)

#### **3.5.5. Teildatensatzübergreifende Regeln**

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

#### **Klassifizierung der teildatensatzübergreifenden Regeln**

Es gibt zwei Arten von teildatensatzübergreifenden Regeln:

- Die Felder sind auf verschiedenen Teildatensätzen eines Moduls definiert.

#### Beispiel:

Regel Modul SA\_HE: GEFDIAGDAT > ENTLDATUM

Das Bogenfeld GEFDIAGDAT ist in Teildatensatz SA\_HE:HI\_TIA, das Bogenfeld ENTLDATUM in SA\_HE:B 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).

#### Beispiel:

Regel Modul SA\_HE: FALLABSCHLUSS <> 9 UND MDS\_GRUND <> LEER

#### **Regeln mit Teildatensatz-Listenfeldern**

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

#### **3.5.6. Verfahren für die Evaluation von Regeln**

Grundsätzlich muss jede gemäß Abschnitt 3.5.3 formulierte Regel evaluiert werden, wenn keine der folgenden Bedingungen zutrifft:

- Für mindestens ein über die Tabelle RegelFelder referenziertes Bogenfeld schlägt eine harte

Feldprüfung fehl. Erst bei Fehlerfreiheit der feldbezogenen Prüfungen werden die feldübergreifenden Prüfungen durchgeführt.

- Ein Feld der Regel ist nicht ausgefüllt (LEER) und ist Bestandteil einer Operation, die einen Wert ungleich LEER erfordert (z.B. FELD > 0). Explizite Vergleiche gegen LEER (Ausdrücke wie FELD = LEER oder FELD <> LEER) sind ausgenommen. Ausdrücke mit Listenoperatoren (EINSIN, KEINSIN, JEDESIN, EINSNICHTIN, IN, NICHTIN) sind ebenso nicht betroffen, da sie auch dann evaluierbar sind, wenn links ein Feld den Wert LEER hat oder rechts eine leere Liste ((LEER) oder ()) steht. Für diese Bedingung gilt folgende Ausnahme: Innerhalb einer Regel steht vor einem Term (z. B. FELD > 0) ein Vergleich desselben Feldes gegen LEER (z.B. FELD = LEER ODER FELD > 0). Dieser Teilausdruck ist auch für den Wert LEER der Variablen FELD zu evaluieren und hat das Ergebnis wahr.
- Eine Funktion der Regel hat das Ergebnis LEER und ist Bestandteil einer Operation, die einen Wert ungleich LEER erfordert.

Teildatensatzübergreifende Regeln (Abschnitt 3.5.5) müssen u. U. mehrfach evaluiert werden (für jede Kombination von Teildatensätzen, welche von der Regel betroffen ist).

### 3.5.7. Feldgruppen

Bogenfelder eines Moduls können zu einer Feldgruppe zusammengefasst werden. Zwischen den Feldern einer Feldgruppe sind logische Abhängigkeiten definiert. In der Praxis bedeutet das, dass der Anwender daran gehindert wird, Felder mit Werten auszufüllen, welche 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 Plausibilitätsregeln, die einen Bezug zu einer Feldgruppe aufweisen (Tabelle Regeln) werden anhand der Feldgruppendefinition automatisch generiert.

Die möglichen Antworten eines jeden Datenfeldes werden in zwei Gruppen aufgeteilt. Die Antworten eines Datenfeldes umfassen hier neben möglichen Werten (z.B. Schlüsselwerten) oder Wertemengen auch die Kategorie „nicht ausgefüllt“ (LEER). Die erste Gruppe ist die Menge der positiven und die zweite Gruppe die Menge der negativen Antworten, in Abhängigkeit der definierten Bedingung eines Feldes in der entsprechenden Feldgruppe.

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 (z.B. 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.

Typen von Feldgruppen

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

### Struktur der Tabellen FeldGruppe und FeldgruppeFelder

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

Struktur der Tabelle FeldGruppe:

| Feldname         | Feldtyp  | Bemerkung   |
|------------------|----------|---|
| idFeldGruppe     | INTEGER  | Primärschlüssel   |
| name             | TEXT(64) | Technischer Name der FeldGruppe   |
| fkModul          | INTEGER  | Obligatorischer Fremdschlüssel zu einem Modul   |
| fkFeldgruppenTyp | INTEGER  | Obligatorischer Fremdschlüssel zu einem Feldgruppentyp  |
| hinweis          | TEXT     | Bei Filter-Feldgruppen relevant für die Gestaltung der Eingabemaske<br>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.<br>Beispiel: Der Hinweistext „Bei postoperativen Komplikationen“ wird oberhalb eines Blocks von Detailkomplikationen angezeigt. |
| fkFilterFeldTyp  | CHAR(1)  | Attribut wird bei Feldgruppen mit mehreren Filterfeldern gesetzt:<br>O = Oder-Verknüpfung der positiven Filterbedingungen<br>U = Und-Verknüpfung der positiven Filterbedingungen  |
| fkRegelTyp       | CHAR(1)  | Fremdschlüssel zur Tabelle RegelTyp:<br>Die Regeltypen sind die in 3.5.1 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D<br>Die generierten Einzelregeln der FeldGruppe haben den gleichen Regeltyp.   |
| nurPositiv       | BIT      | Nur bei Filter-Feldgruppen wirksam:<br>Wenn wahr, dann umfasst die FeldGruppe nur diejenigen Regeln, welche sich auf die positive Filterbedingung beziehen  |
| grauWennNegativ  | BIT      | Definiert eine Layout-FeldGruppe, siehe im Abschnitt weiter unten.  |

Struktur der Tabelle FeldgruppeFelder:

| Feldname                   | Feldtyp | Bemerkung  |
|----------------------------|---------|--|
| idFeldgruppeFelder         | INTEGER | Primärschlüssel  |
| fkFeldGruppe               | INTEGER | Obligatorischer Fremdschlüssel zur Feldgruppe  |
| fkBogenFeld                | INTEGER | Obligatorischer Fremdschlüssel zum Bogenfeld   |
| bedingung                  | TEXT    | Positive Bedingung für das jeweilige Bogenfeld   |
| 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. |
| tdsFilter                  | BIT     | Das Bogenfeld wird in Regeln als TDS-Listefeld 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 (Abschnitt 3.5.4) verwendet werden. Die auf der rechten Seite erlaubten Operanden sind nachfolgend aufgelistet:

- Literale
- LEER
- Kodelisten, in denen auch die Codes eines Schlüssels referenziert werden können, z.B. (1;2;3)
- ICD-Listen oder OPS-Listen (z. B. SA\_ICD)

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

Beispiele für Bedingungen von Feldgruppen:

- MDS\_GRUND <> LEER
- BILDDURCH IN (1;2)

### 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, welches mit F bezeichnet wird. Dann lässt sich eine Feldgruppe in folgender Tabelle darstellen:

Formale Definition einer Feldgruppe:

| Feld | Positive Bedingung | Negative Bedingung | Bemerkung |
|------|--------------------|--------------------|-----------|
|------|--------------------|--------------------|-----------|

| F              | p(F)               | n(F)               | falls Feldgruppentyp mit Filter |
|----------------|--------------------|--------------------|---------------------------------|
| A <sub>1</sub> | p(A <sub>1</sub> ) | n(A <sub>1</sub> ) |                                 |
| A <sub>2</sub> | p(A <sub>2</sub> ) | n(A <sub>2</sub> ) |                                 |
| A <sub>3</sub> | p(A <sub>3</sub> ) | n(A <sub>3</sub> ) |                                 |
| ...            |                    |                    |                                 |
| A <sub>n</sub> | p(A <sub>n</sub> ) | n(A <sub>n</sub> ) |                                 |

Eine Feldgruppe bestehe insgesamt aus n abhängigen Bogenfeldern:

A<sub>1</sub>, A<sub>2</sub>, ... A<sub>n</sub>

In Abhängigkeit von den Feldgruppentypen werden unterschiedliche Einzelregeln generiert.

### Feldgruppen mit Filter

- Regeln der Feldgruppe "Optionale Mehrfachauswahl mit Filterfeld" (MF\_OPTIONAL\_FILTER)

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

Insgesamt sind n Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe "Obligatorische Mehrfachauswahl mit Filterfeld" (MF\_MINDESTENS1\_FILTER)

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

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

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)

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

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

Insgesamt sind 2n Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe "Einfachauswahl mit Filter" (EF\_FILTER)

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

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

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

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

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

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

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

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

### Achtung:

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:

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

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

### **Feldgruppen ohne Filter**

- Regeln der Feldgruppe "Einfachauswahl" (EF)

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

$p(Aj) \text{ UND } p(Ai)$  für alle unterschiedlichen  $i, j = 1, \dots, n$

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

- Regeln der Feldgruppe "Obligatorische Mehrfachauswahl" (MF\_MINDESTENS1)

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

Insgesamt ist eine Einzelregel mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe "Und-Regel" (UND)

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

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  $F1, F2, \dots, Fn$  mit den positiven bzw. negativen Bedingungen  $p(Fj)$  bzw.  $n(Fj)$ . Für alle Filterfelder wird eine positive Bedingung  $p(F1, \dots, Fn)$  und eine negative Bedingung  $n(F1, \dots, Fn)$  gebildet. Diese modifizierten Filterbedingungen ersetzen die 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.

### **Gestaltung von Eingabemasken mit Feldgruppen**

|                  |  |
|------------------|--|
| 39-47            | <b>Thrombolyse/Rekanalisation</b>  |
| 39               | <b>Systemische Thrombolyse</b> <input type="checkbox"/><br>0 = nein<br>1 = Verlegung zur Thrombolyse<br>2 = ja, in vorbehandelnder Einrichtung<br>3 = ja, im eigenen Haus  |
| wenn Feld 39 = 3 |  |
| 40>              | <b>Beginn Lyse am</b><br>TT.MM.JJJJ<br><input type="text"/> <input type="text"/> . <input type="text"/> <input type="text"/> . <input type="text"/> <input type="text"/> <input type="text"/> <input type="text"/> |
| 41>              | <b>Beginn Lyse um</b><br>HH:MM<br><input type="text"/> : <input type="text"/>  |

Feldgruppen, bei denen das Attribut grauWennNegativ in der Datenbanktabelle FeldGruppe gesetzt ist, sollen für die Gestaltung von Eingabemasken verwendet werden und werden nachfolgend auch als Layout-Feldgruppen bezeichnet. Der Attributname grauWennNegativ wurde gewählt, weil die abhängigen Felder der Layout-Feldgruppen auf den generierten Dokumentationsbögen eingegraut sind. Die linke Abbildung zeigt die Layout-Feldgruppe SA\_HE:LyselmEigenenHaus mit dem Filterfeld 39 „Systematische Thrombolyse“ und den abhängigen Feldern 40 „Beginn Lyse am“ und 41 „Beginn Lyse um“, die ausgefüllt werden müssen, wenn Feld 39 = 3 ist und sonst leer bleiben.

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 nein.
- Bei Vorliegen dieser drei Eigenschaften müssen die abhängigen Felder leer bleiben, wenn die negative Filterbedingung bei der Dokumentation eines Falles erfüllt ist.

#### Beispiel:

Wenn in Datenfeld 39 <> 3 (ja, im eigenen Haus) angegeben ist, so müssen die Datenfelder 40 und 41 leer bleiben. Die folgenden Plausibilitätsprüfungen stellen dieses sicher:

| bedingung                              | meldung   | fkFeldGruppe            |
|--|---|-------------------------|
| SYSTTHLYSE <> 3 UND THLYSEDAT <> LEER  | "Systemische Thrombolyse" ist nicht "ja, im eigenen Haus" [3], obwohl "Beginn Lyse am" ausgefüllt ist | SA_HE:LyselmEigenenHaus |
| SYSTTHLYSE <> 3 UND THLYSEZEIT <> LEER | "Systemische Thrombolyse" ist nicht "ja, im eigenen Haus" [3], obwohl "Beginn Lyse um" ausgefüllt ist | SA_HE:LyselmEigenenHaus |

### Empfehlung zur Umsetzung von Layout-Feldgruppen

Ist dieses Attribut 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.

Achtung: Deaktivierte Felder dürfen nicht versteckt werden.

### 3.5.8. Leitlinien für die Gestaltung der Benutzeroberflächen von Erfassungsprogrammen

Die Grundsätze für die Plausibilitätsprüfungen wirken sich insbesondere auf die Gestaltung der Benutzeroberflächen in Erfassungsprogrammen aus. Die nachfolgend vorgestellten Regeln sind

Ergebnis der Empfehlungen der Fachgruppen sowie der Arbeitsgruppe „Plausibilitätsprüfungen“. Grundsätzlich sollte durch die funktionale Gestaltung der Benutzeroberfläche 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. Externe Systeme, welche Daten an ein Erfassungsprogramm übergeben, sollten diese Grundsätze sinngemäß anwenden.

- Keine Suggestion von Feldinhalten durch Vorbelegung (Defaults). Bei wenigen Feldern (Felder, IKNRKH, STANDORT, BSNR, FACHABT) und bei den Zusatzfeldern darf von dieser Regel abgewichen werden. 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.
- Verwendung der vorgeschriebenen Fehler- und Warnmeldungen bei feldübergreifenden Regeln: Die Fehler- und Warnmeldungen sind so formuliert, dass sie möglichst nicht suggerieren, auf welche Weise 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 solchen 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 der Benutzerakzeptanz abträglich ist, wenn diese Systematik auch an allen anderen Stellen durchgängig verfolgt wird.
- Kein automatisches Verändern von Feldinhalten in Abhängigkeit von anderen Feldinhalten: Ein Beispiel: wenn die „Bildgebung durchgeführt“ zunächst auf 2 (1. Bildgebung im eigenen Haus) eingestellt wurde und die Felder „Bildgebung durchgeführt am/um“ ausgefüllt worden sind, soll das Ändern des Wertes des übergeordneten Feldes „Bildgebung durchgeführt“ nicht automatisch zum Löschen der Inhalte bei „Bildgebung durchgeführt am/um“ führen. Vielmehr soll eine Fehlermeldung erfolgen und der Anwender gezwungen sein, zunächst die Felder „Bildgebung durchgeführt am/um“ zu leeren, bevor der Wert in „Bildgebung durchgeführt“ geändert werden kann.

### 3.6. Listen von Schlüsselkodes (OPS, ICD-10-GM)

In der Spezifikation sind Listen von OPS- bzw. ICD-10-GM-Kodes in separaten Tabellen definiert. Die aktuell gültigen Kataloge sind über das DIMDI (<http://www.dimdi.de>) zu beziehen. Jede Liste hat einen technischen Namen (z. B. MRE\_ICD) und eine erläuternde Bezeichnung (z. B. „Erreger mit bestimmten Antibiotikaresistenzen“). Die Listen sind in insgesamt vier Tabellen der Spezifikationsdatenbank definiert. Die technischen Namen und erläuternden Bezeichnungen sind in den Tabellen OPSListe bzw. ICDListe definiert.

Die Codes finden sich in den Tabellen OPSWert bzw. ICDWert.

#### Achtung:

Einige Listen der Tabellen OPSListe oder ICDListe sind als QS-Filter-Listen deklariert (Attribut qsFilter). Die Definitionen der QS-Filter-Listen sind in der für das Erfassungsjahr gültigen Spezifikation für QS-Filter zu finden. Dort existieren die gleichen Tabellen OPSListe, ICDListe, OPSWert und ICDWert mit den entsprechenden Listendefinitionen.

#### Beispiel:

Die Einschlussdiagnosen Schlaganfall Hessen (SA\_ICD) sind in der Spezifikation für QS-Filter zu finden.

### 3.6.1. OPS-Listen

Jede OPS-Liste ist charakterisiert durch ihren Namen (Attribut name in Tabelle OPSListe), welcher nach folgendem Schema gebildet wird:

$\{ \langle \text{TEXT} \rangle \} \text{OPS} \{ \_ \langle \text{TEXT} \rangle \}$

Hinter  $\langle \text{TEXT} \rangle$  verbirgt sich ein frei wählbarer Name (Erlaubte Zeichen: A–Z, a–z, 0–9, \_, Umlaute sind nicht erlaubt). Die {}-Ausdrücke sind optional.

#### Umgang mit Seitenlokalisationen

Die Codes der OPS-Listen enthalten keine Seitenlokalisationen, obwohl die Zusatzkennzeichen für Seitenbezeichnung R, L oder B für Prozeduren an Lokalisationen, die paarig vorhanden sind (z. B. Leiste, Niere, Oberschenkel etc.) verpflichtend zu dokumentieren sind. In der QS-Dokumentation wird das Zusatzkennzeichen für die Seitenbezeichnung getrennt durch einen Doppelpunkt dem OPS-Code angehängt. Fehlt ein erforderliches Zusatzkennzeichen, so ist die Dokumentation unplausibel.

#### Beispiel:

Für den OPS-Code 5-144.01 (Extrakapsuläre Extraktion der Linse [ECCE]: Über sklerokornealen Zugang: Mit Einführung einer kapselfixierten Hinterkammerlinse = Einschlussprozedur für den Datensatz Kataraktchirurgie) ist eine Seitenangabe erforderlich. Daher sind folgende Codes gültig:

- 5-144.01:R
- 5-144.01:L
- 5-144.01:B

Für die Prüfung, ob zwei Codes identisch sind, genügt kein einfacher Stringvergleich. Stattdessen wird ein Stringvergleich der Normcodes durchgeführt, um die Übereinstimmung zwischen dokumentiertem Code und dem Code einer OPS-Liste zu ermitteln. Jeder OPS-Code lässt sich entweder als Code mit Seitenlokalisation:  $\text{code} = \text{normCode} + \text{seite}$  oder als Code ohne Seitenlokalisation  $\text{code} = \text{normCode}$  darstellen.

Identitätsprüfung zwischen dokumentierten OPS-Kodes und Codes von OPS-Listen:

| Dokumentierter OPS-Code                                    | OPS-Code der OPS-Liste           | Bedingung für Gleichheit (= ist Stringvergleich)                |
|--|----------------------------------|---|
| $\text{normCode}_{\text{Dok}} + \text{seite}_{\text{Dok}}$ | $\text{normCode}_{\text{Liste}}$ | $\text{normCode}_{\text{Dok}} = \text{normCode}_{\text{Liste}}$ |
| $\text{normCode}_{\text{Dok}}$                             | $\text{normCode}_{\text{Liste}}$ | $\text{normCode}_{\text{Dok}} = \text{normCode}_{\text{Liste}}$ |

### 3.6.2. ICD-Listen

Jede ICD-Liste ist charakterisiert durch ihren Namen (Attribut name in Tabelle ICDListe), welcher per definitionem folgendem Schema gehorcht:

$\{ \langle \text{TEXT} \rangle \_ \} \text{ICD} \{ \_ \langle \text{TEXT} \rangle \}$

Hinter  $\langle \text{TEXT} \rangle$  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 Codes 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.

**Achtung:** Die Suffixe \*, +, ! entfallen in der Spezifikation! Es ist zu beachten, dass im Krankenhaus dokumentierte ICD-Kodes die Suffixe \*, +, ! enthalten können.

## 3.7. Versionierung

### 3.7.1. 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 Erfassungsjahr (z. B. 1. Januar 2024 bis 31. Dezember 2024).

#### Achtung:

Die QS-Dokumentationssoftware eines Erfassungsjahres wird für diejenigen Behandlungsfälle verwendet, deren Aufnahmedatum ins Krankenhaus in den oben definierten Gültigkeitszeitraum fällt. Bei sogenannten „Überliegern“ (Aufnahmedatum im alten Jahr, Entlassungsdatum im nachfolgenden Jahr) wird die QS-Dokumentationssoftware auch noch für Behandlungsfälle benutzt, welche nach dem in der Datenbank definierten Gültigkeitszeitraum (in der Regel nach dem 31. Dezember) entlassen worden sind.

Jedes Modul der Datenbank hat eine Version (vgl. Attribut fkVersion in Tabelle Modul). Über die in der Datenbank definierten Relationen sind auch für alle Bogenfelder (Tabelle BogenFeld), Exportfelder (Tabelle ExportFormat) und Plausibilitätsregeln (Tabelle Regeln) Versionen definiert.

#### **Finale Spezifikation – Zwischenstände**

Versionen können den Status „in Entwicklung“ oder „final“ haben. Diese Zustände werden in der Nachschlagetabelle VersStatus verwaltet. Das Attribut gueltig zeigt die gültige Version der Datenbank an. Es darf nur eine einzige Version als gültig markiert sein.

#### **Historie der Versionen**

Die Tabelle Version enthält auch einen Selbstbezug (Attribut fkVersion), welche die Identifizierung der Vorgängerversion ermöglicht.

#### Beispiel:

Die Vorgängerversion der Spezifikation 2024 V02 ist die Version 2024 V01.

### 3.7.2. Delta-Informationen zur vorhergehenden Version

Um den Nutzern der Spezifikation umfassende Informationen zu den jeweiligen Änderungen zur Verfügung zu stellen, enthält die neue Spezifikationsdatenbank Tabellen, die den Änderungsstand zur letzten Version der Datenbank anzeigen.

Es gibt drei Delta-Tabellen, welche 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 ist die Zuordnung zu den Entitäten der Datenbank möglich.

Struktur der Tabelle DeltaNeu:

| Feldname           | Feldtyp  | Bemerkung   |
|--------------------|----------|---|
| idDeltaNeu         | AutoWert | Primärschlüssel                                       |
| id                 | Text     | ID der Entität, die eingefügt wurde                   |
| fkTabellenStruktur | Zahl     | Bezug zur Tabelle, in der die Entität eingefügt wurde |
| bemerkung          | Memo     | Begründung für die Ergänzung                          |

#### Beispiel:

Die Zeile idDeltaNeu = 2001 enthält folgende Werte:



id = 2089

fkTabellenStruktur = <Regeln> bemerkung = KLAS\_ICD NICHTIN SA\_ICD

Dies bedeutet, dass in der Tabelle Regeln ein neuer Eintrag mit idRegeln = 2089 angelegt worden ist. Unter bemerkung finden sich weitere Erläuterungen zum neuen Tabelleneintrag.

### Geänderte Entitäten

Über die Attribute id und fkTabellenFeldStruktur der Tabelle DeltaAttribut ist die Zuordnung zu den Attributen der Entitäten der Datenbank möglich.

Struktur der Tabelle DeltaAttribut:

| Feldname               | Feldtyp  | Bemerkung  |
|------------------------|----------|--|
| idDeltaAttribut        | AutoWert | Primärschlüssel  |
| id                     | Text     | ID der Entität, die geändert wurde                                       |
| fkTabellenFeldStruktur | Zahl     | Bezug zum Attribut einer Tabelle, in der die Entität geändert wurde      |
| alterInhalt            | Memo     | Alter Inhalt der geänderten Entität in der letzten finalen Spezifikation |
| neuerInhalt            | Memo     | Neuer Inhalt dieser Entität in der aktuellen Spezifikation               |
| bemerkung              | Memo     | Begründung für die Ergänzung   |

#### Beispiel:

Die Zeile idDeltaAttribut = 22 enthält folgende Werte:

id = 12

Tabellenattribut = <FeldGruppeFelder.bedingung>

alterInhalt = 1

neuerInhalt = IN (1;2)

Das heißt, dass in der Zeile idFeldGruppeFelder = 12 der Tabelle FeldGruppeFelder das Attribut bedingung von „1“ auf „IN (1;2)“ geändert wurde. Die Spalte bemerkung enthält ggf. weitere Hinweise zum Kontext der Änderung.

### Gelöschte Entitäten

Über die Attribute id und fkTabellenStruktur in der Tabelle DeltaGeloesch ist die Zuordnung zu den Entitäten der Datenbank möglich.

Struktur der Tabelle DeltaGeloesch:

| Feldname           | Feldtyp  | Bemerkung   |
|--------------------|----------|---|
| idDeltaGeloesch    | AutoWert | Primärschlüssel   |
| id                 | Text     | ID der Entität, die gelöscht wurde                                  |
| fkTabellenStruktur | Zahl     | Bezug zum Attribut einer Tabelle, in der die Entität gelöscht wurde |
| bemerkung          | Memo     | Begründung für die Ergänzung  |

#### Beispiel:

Die Zeile idDeltaGeloesch = 2003 enthält folgende Werte:

Id = 2004

fkTabellenStruktur = <ICDListe>

bemerkung="TIA\_ICD"

Das heißt, dass aus der Tabelle ICDListe der Vorgängerdatenbank der Eintrag in ICDListe mit der id = 2004 gelöscht wurde. Unter bemerkung finden sich ggf. weitere Hinweise zum Kontext der gelöschten Entität.

### 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 = Ja gesetzt ist.

Inhalt der Tabelle TabellenFeldStruktur:

| idTabelle | feldName          | ... | deltaAktiv | fkTabellenFeldStruktur |
|-----------|-------------------|-----|------------|------------------------|
| 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.

### 3.7.3. Abgrenzung zwischen Erfassungsjahren und Datensatzformaten

Die LAGQH muss Datensätze von Krankenhäusern entgegennehmen, falls die Aufnahmedaten des Krankenhauses in den Gültigkeitszeitraum der Version einer finalen Datenbank fallen. Eine Datenbank ist dann final, wenn alle ihre Module die gleiche finale Version haben und diese finale Version in der Tabelle Version als gültig markiert ist.

Das Abgrenzungskriterium zwischen den Erfassungsjahren ist das Datum der Aufnahme in das Krankenhaus (AUFNDATUM) bei Datensätzen stationär aufgenommener Patienten

#### Achtung:

Dem Erfassungsjahr 2024 zugeordnete Fälle müssen im Format der Spezifikation 2024 an die LAGQH gesandt werden, sonst ist die Datenlieferung zurückzuweisen.

#### Hinweis:

Für Fälle, die bereits abgeschlossen und an die Datenannahmestelle übermittelt wurden, sollte nach Ablauf der Datenlieferfrist kein erneuter Export in Form eines Updates oder Stornos möglich sein.

### 3.7.4. Version des Exportverfahrens

Das Attribut fkGueltigExportVerfahren gibt an, welche Versionskürzel (Attribut Version.name) in der ersten Zeile der Steuerdatei verwendet werden. Nur eine Version darf als gültig für das Exportverfahren markiert sein. Da das Exportverfahren nicht jährlich geändert wird, kann die Version des Exportverfahrens der aktuellen Version der Spezifikation „hinterherhinken“.

Für das Erfassungsjahr 2024 gilt das Exportverfahren 2024.

### **3.7.5. Version der Exportdateien**

Die Version der Exportdateien entspricht dem Jahr der Spezifikation.

## 4. Export

### 4.1. Datenexport

Die erhobenen Datensätze der hessischen Leistungsbereiche sind von allen Krankenhäusern unverzüglich gemäß dem vorgegebenen Datenexportformat der von der LAGQH zur Verfügung zu stellen. Dabei wird das Verfahren QSINDIREKT angewandt.

Für jeden Teildatensatz eines Moduls wird vom Dokumentationssystem eine eigene Exportdatei generiert, welche eine Kopfzeile mit den Feldnamen und nachfolgend die exportierten Datensätze enthält (ASCII-Format mit Semikolon als Trennzeichen). Die Struktur einer Exportdatei orientiert sich – unter Berücksichtigung der Pseudonymisierungsvorschriften – an der Datenfeldbeschreibung des Teildatensatzes und wird um die Zusatzfelder ergänzt.

Aus einer Steuerdatei und einer oder mehreren Exportdateien wird vom Dokumentationssystem eine komprimierte und verschlüsselte Transaktionsdatei für den Versand von abgeschlossenen Vorgängen (Datensätzen) an die entgegennehmende Stelle erzeugt.

#### 4.1.1. Registrierung eines Dokumentationssystems

Die Registrierung eines Dokumentationssystems bei der LAGQH ist Voraussetzung für die Datenübermittlung. Es dürfen nur solche Dokumentationssysteme, welche bei der LAGQH registriert sind, Datensätze (Qualitätssicherungsverfahren QSINDIREKT) übermitteln.

Registrierte Dokumentationssysteme bekommen von der LAGQH eine Registriernummer zugewiesen, welche folgendermaßen aufgebaut ist.

`<Registriernummer> = <Länderkode><Registrierkode>`

Jedes Dokumentationssystem im Krankenhaus ist somit bundesweit identifizierbar. Beim Wechsel eines Dokumentationssystems innerhalb eines Krankenhauses ist zu beachten, dass eine neue Registriernummer zu beantragen ist.

Beispiel:

HE1234a – Länderkode HE für Hessen + Registrierkode, 5-stellig

#### 4.1.2. Identifizierung von Datensätzen

Die Vorgangsnummer kennzeichnet in eindeutiger Weise jeden dokumentierten Datensatz eines Dokumentationssystems, und zwar unabhängig vom angewandten Modul.

Im einfachsten Fall könnte also die Vorgangsnummer um 1 erhöht werden, wenn ein neuer Datensatz angelegt wird. Wenn während eines stationären Aufenthaltes zwei QS-Dokumentationen eines Krankenhausfalls angelegt werden, so müssen auch unterschiedliche Vorgangsnummern vergeben werden.

Keinesfalls zulässig ist es, eine Patientenidentifikationsnummer oder die Krankenhaus-Fallnummer zu verwenden bzw. zu pseudonymisieren, denn für jeden Patienten können unter Umständen mehrere Datensätze angelegt werden. Bei der Umsetzung hat der Softwareanbieter weitgehende Freiheit, vorausgesetzt, dass die modulübergreifende Eindeutigkeit der Vorgangsnummer gewährleistet ist.

Die Vorgangsnummer, welche als Zusatzfeld mit jedem Teildatensatz an die entgegennehmende Stelle übermittelt wird, darf für die Datenstelle nicht personenbeziehbar sein. In der Vorgangsnummer darf z. B. nicht das Geburtsdatum enthalten sein.

Die QS-Dokumentationssoftware verwaltet jahrgangsübergreifend die Vorgangsnummern der QS-Dokumentationen. Sie soll dem Krankenhaus eine Zuordnung der Vorgangsnummern zum krankenhausinternen Fall oder Patientennummern (vgl. nicht übermitteltes Datenfeld IDNRPAT) ermöglichen.

### **Annahme oder Ablehnung von unterschiedlichen Versionen eines Datensatzes**

Bei der entgegennehmenden Stelle eingehende Datensätze werden anhand der Kombination aus Registriernummer und Vorgangsnummer als ein Vorgang identifiziert. Der für den Vorgang gespeicherte Datensatz kann nur durch eine neuere Version (mit höherer Versionsnummer) überschrieben werden.

Unterschiedliche Versionen eines Datensatzes müssen demselben Primärmodul zugeordnet sein. Jedem bei der entgegennehmenden Stelle eingehenden Datensatz ist ein Primärmodul (Definition siehe unten) zugeordnet.

Findet ein Wechsel des Primärmoduls statt, so ist ein neuer Datensatz immer abzulehnen. Ebenso abgelehnt wird ein Datensatz, wenn bereits eine aktuellere Version (höhere Versionsnummer) übermittelt wurde.

### **4.1.3. Der Exportvorgang**

Bei einem Exportvorgang wird von einem Dokumentationssystem eine Transaktionsdatei für die LAGQH erzeugt.

Die Daten der zu exportierenden QS-Dokumentationen werden vom Dokumentationssystem in Exportdateien geschrieben und die entsprechenden Vorgänge (identifiziert durch Vorgangsnummern) im Dokumentationssystem als „exportiert“ markiert. Jeder Exportvorgang erfolgt gemäß dem jeweils gültigen Exportverfahren.

Jeder Übermittlung an die LAGQH ordnet das registrierte Dokumentationssystem eine eindeutige Transaktionsnummer zu. Die Eindeutigkeit der Transaktionsnummer muss innerhalb eines Erfassungsjahres hinweg gewährleistet sein.

#### Achtung:

Nicht zulässig ist es, für den erneuten Export korrigierter Datensätze die gleiche Transaktionsnummer wie für den ersten Export zu verwenden.

Die Transaktionsnummer ist als eindeutige, fortlaufende, positive und ganze Zahl zu wählen: 1, 2, 3 etc.

### **Die Steuerdatei**

Die Steuerdatei – das Inhaltsverzeichnis eines Exportvorgangs – wird nach folgendem Schema benannt:

HEADER.<Transaktionsnummer>

#### Beispiel:

HEADER.1 HEADER.001 HEADER.0815 HEADER.88815

Die erste Zeile der Steuerdatei besteht aus folgenden Feldern:

- Verfahrenskennung QSINDIREKT,
- Jahr der Spezifikation (Tabelle Version),
- Kennung der Software, mit der die Exportdateien erzeugt wurden,
- Datum und Uhrzeit der Erstellung der Exportdatei,
- Institutionskennzeichen des Absenders (Krankenhaus),
- Registriernummer des Dokumentationssystems (Abschnitt 4.1.1) und

- Namenszeichen des Ansprechpartners zu Fragen der Datenübermittlung.

Jede nachfolgende Zeile spezifiziert eine Exportdatei und setzt sich aus folgenden Feldern zusammen:

- Modulbezeichnung (Attribut name in Tabelle Modul)
- Name des Teildatensatzes (Attribut name in Tabelle Bogen)
- Jahr der Spezifikation (Tabelle Version)
- Name der Datei (zurzeit fest vorgegeben) und
- Anzahl der Datensätze in diesem Modul.

Die Felder sind durch Semikola voneinander getrennt. Jede Zeile wird durch <CR><LF> (ASCII 13, 10) abgeschlossen.

#### Beispiel:

Steuerdatei des Verfahrens QSINDIREKT:

```
QSINDIREKT;2024;2024 V03#QS-Doc#2.31;11.11.2024 11:11:11;260612345;HE1234A;NN<CR><LF>
SA_HE;B;2024;MSA_HEB.001;15<CR><LF>
SA_HE;HI_TIA;2024;MSA_HEHI_TIA.001;11<CR><LF>
SA_HE;ICB_SAB;2024;MSA_HEICB_SAB.001;3<CR><LF>
MRE_HE;B;2024;MMRE_HEB.001;22<CR><LF>
MRE_HE;MRSA;2024;MMRE_HEMRSA.001;22<CR><LF>
```

**Achtung:** Es ist nicht zulässig, Exportdateien unterschiedlicher Verfahrenskennungen und unterschiedlicher Versionen in einer einzigen Transaktion zu übermitteln.

### **Softwarekennung**

Die Softwarekennung setzt sich aus folgenden Teilen zusammen:

- Version der Spezifikation, auf deren Basis die QS-Dokumentationssoftware entwickelt wurde (Attribut name in Tabelle Version)
- Name der QS-Dokumentationssoftware
- Version der QS-Dokumentationssoftware

Die Teile der Softwarekennung sind durch das Zeichen # voneinander getrennt:

<Version>#<Name Software>#<Version Software>

#### Beispiele:

2024 V03#QSDOC#2.31

Exporte, deren Softwarekennung nicht dem neuen Format genügt, sind fehlerhaft (Fehlerart STEUER).

Standardisierter Fehlertext:

Softwarekennung in Steuerdatei fehlerhaft: Angabe im Format <Spezversion>#<Name Software>#<Version Software> erforderlich!

### **Erzeugung der Transaktionsdatei**

Die Steuerdatei und die zugehörigen Exportdateien werden vor dem Export mit Hilfe eines pkzip 2.04g-kompatiblen Archivierungsprogramms komprimiert und anschließend mit Triple-DES verschlüsselt.

Das Krankenhaus erhält bei der Registrierung seiner Dokumentationssysteme von der LAGQH einen Verschlüsselungskode, der bei der Erzeugung der Archivdatei verwendet werden muss. Die Transaktionsdatei wird wie folgt benannt:

T-<Länder-Kode><Registrierkode>-<Transaktionsnummer>.ZIP.3DES

Beispiel:

T-HE1234A-001.ZIP.3DES

#### 4.1.4. Export von Teildatensätzen

Beim Export einer QS-Dokumentation durch ein Dokumentationssystem werden die Inhalte der für den betreffenden Vorgang angelegten Teildatensätze in passende Exportdateien geschrieben. Jede Exportdatei ist einem in der Spezifikation definierten Teildatensatz (vgl. Tabelle Bogen) zugeordnet.

##### Pseudonymisierung

Für die Übermittlung der Qualitätssicherungsdaten an die LAGQH müssen personenbeziehbare Daten in manchen anderen Bundesländern ausreichend pseudonymisiert werden. Die Pseudonymisierung betrifft dort z.B. alle Kalendertagesdaten.

Sie werden z. B. umgerechnet in die Differenz in Tagen zwischen dem Kalendertagesdatum und dem AUFNDATUM. Die Differenzen sind positiv anzugeben. Die Vorschriften der Pseudonymisierung von Bogenfeldern findet man in den Tabellen ErsatzFeld und ErsatzFuerFeld.

##### Ersatzfelder für den Datenexport

Ersatzfelder werden aus ein oder mehreren Feldern der Datenfeldbeschreibung berechnet. Mit einem Ersatzfeld verknüpfte Bogenfelder werden nicht exportiert, wenn sie nicht als <bleibt> gekennzeichnet wurden. Stattdessen werden ein oder mehrere Ersatzfelder exportiert. Vorrangig dienen Ersatzfelder der Pseudonymisierung beim Datenexport. Die verwendeten Ersatzfelder sind in der Tabelle ErsatzFeld gespeichert.

Struktur der Tabelle ErsatzFeld:

| Feldname     | Feldtyp | Bemerkung                                   |
|--------------|---------|---|
| idErsatzFeld | INTEGER | Primärschlüssel                             |
| name         | TEXT    | Technischer Name (muss eindeutig sein)      |
| bezeichnung  | TEXT    | Beschreibender Text                         |
| formel       | TEXT    | Berechnungsformel der Ersatzfelder          |
| fkBasisTyp   | INTEGER | Obligatorischer Fremdschlüssel zum Basistyp |
| fkSchluessel | INTEGER | Optionalen Fremdschlüssel zum Schlüssel     |

Beispiel:

Das Ersatzfeld Verweildauer wird folgendermaßen berechnet:

$$\text{vwdauer} = \text{ENTL DATUM} - \text{AUFN DATUM}$$

Die Pseudonymisierung von Datenfeldern wird in der Tabelle ErsatzFuerFeld konfiguriert. Für die Programmierung der Exportfelder ist dieser Abschnitt weniger wichtig, da die Exportfelder direkt über die Abfrage ExportFelderFürEinModul 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 Pseudonymisierung ist die Standardpseudonymisierung für alle Module. Sie kann jedoch durch eine modulspezifische Pseudonymisierung überschrieben werden: Wenn ein Ersatzfeld mit einem Bogenfeld (über Attribut fkBogenFeld) verknüpft ist, so wird statt des Bogenfeldes das berechnete Ersatzfeld in die Exportdatei des Teildatensatzes geschrieben

Struktur der Tabelle ErsatzFuerFeld:



| Feldname         | Feldtyp | Bemerkung  |
|------------------|---------|--|
| idErsatzFuerFeld | INTEGER | Primärschlüssel  |
| fkFeld           | INTEGER | Optionaler Fremdschlüssel zur Tabelle Feld   |
| fkBogenFeld      | TEXT    | Optionaler Fremdschlüssel zur Tabelle BogenFeld  |
| fkErsatzFeld     | TEXT    | Obligatorischer Fremdschlüssel zur Tabelle ErsatzFeld  |
| parametrierbar   | BOOL    | Gibt eine parametrierbare Verknüpfung zwischen Feld bzw. Bogenfeld und Ersatzfeld an (WAHR = parametrierbar) |

#### Beispiele:

- Das Feld ENTLDATUM (Modul SA\_HE) hat die Ersatzfelder poopvwdauer, entlquartal und entlwochtag.
- Wenn das Ersatzfeld <entfällt> mit einem Bogenfeld verknüpft ist, so entfällt das Bogenfeld in der Exportdatei.
- Wenn das Ersatzfeld <bleibt> mit einem Bogenfeld verknüpft ist, so wird das Bogenfeld unverändert in die Exportdatei übernommen. Dieses Ersatzfeld ist nur als parametrierbare Verknüpfung sinnvoll (s. u.).

Was ist zu tun, wenn ein einziges Ersatzfeld sowohl über fkFeld als auch über fkBogenFeld definiert ist („doppelte Definition“)?

In diesem Fall hat die spezielle Pseudonymisierung (über fkBogenFeld) Vorrang, die allgemeinen Pseudonymisierungen (fkFeld) werden ignoriert. Die allgemeine Definition kommt nur in den Modulen zur Anwendung, in denen keine spezielle Pseudonymisierung vorliegt.

### **Parametrierung**

Die Verknüpfung zwischen Feld bzw. Bogenfeld und Ersatzfeld kann parametrierbar werden (Attribut parametrierbar). Parametrierbare Ersatzfelder erscheinen immer als eigene Spalte 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 der Landesebene berücksichtigt werden. In Hessen z.B. dürfen Postleitzahlen und Datumsfelder unverändert exportiert werden, in anderen Bundesländern gibt es schärfere Anforderungen, sodass nicht alle Stellen einer Postleitzahl exportiert werden.

### **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:

- 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, so wird kein Wert für das Ersatzfeld exportiert (bleibt LEER).

### **Aufbau der Exportdatei**

Die Struktur der Exportdateien ergibt sich im Wesentlichen aus der Datenfeldbeschreibung der einzelnen Module und den oben beschriebenen Pseudonymisierungsmaßnahmen zur Erfüllung der datenschutzrechtlichen Voraussetzungen.

Die Exportdateien sind wie folgt aufgebaut:

- ASCII-Format, erweiterter IBM-PC-Zeichensatz (8 Bit),
- die erste Zeile enthält die Feldnamen, die nachfolgenden Zeilen enthalten die Werte der Exportfelder eines Teildatensatzes,

- Zeilen werden durch <CR><LF> (ASCII 13, 10) abgeschlossen,
- Feldnamen in der ersten und Werte der Exportfelder in den folgenden Zeilen werden durch Semikola getrennt und können variabel lang sein.

### Felder der Exportdatei

Einen Überblick über die zu exportierenden Felder eines Moduls liefert die Abfrage ExportFelderFürEinModul. Am Anfang eines jeden Teildatensatzes findet man die Zusatzfelder. Danach folgen die Bogenfelder des Teildatensatzes, wobei die zu pseudonymisierenden Bogenfelder durch die Ersatzfelder ersetzt sind.

### Format der Exportfelder

In Abhängigkeit vom Basistyp des Exportfeldes werden die Werte der Exportfelder exportiert. Die Basistypen, deren Literale in Regeln von doppelten Anführungszeichen oder Hochkommata umschlossen werden, werden ohne diese Zeichen exportiert (TEXT, SCHLUESSEL, DATUM, MONDATUM, QUARTDATUM und UHRZEIT). Die Formatanweisungen entsprechen (bis auf die Anführungszeichen und Hochkommata) den in Abschnitt 3.5.4 aufgelisteten Beispielen.

Folgende Grundsätze für das Format von Werten der Exportfelder sind nachfolgend noch einmal explizit aufgelistet:

- Semikola dürfen innerhalb eines Exportfeldes nicht vorkommen,
- Dezimaltrennzeichen ist das Komma, es werden keine Tausendertrennzeichen verwendet,
- Datumentrennzeichen ist der Punkt (Datumsangaben sind immer 10-stellig),
- Uhrzeittrennzeichen ist der Doppelpunkt,
- Die Codes nach ICD-10-GM und OPS werden mit Trennzeichen übermittelt.

#### Beispiel:

Dieses Beispiel zeigt den Anfang einer Exportdatei für das Modul MRE\_HE. Es sind nur die ersten 11 Felder dargestellt:

```
RegistrierNr;Vorgangsnr;VersionNr;Storno;Modul;Bogen;DokAbschlDat;IKNRKH;standortID
;BSNR;FACHABT;IDNRPAT;.. <CR><LF>
HE1234A;202401001;1;;MRE_HE;B;31.01.2024;260612345;01;1;1700;08154711;.. <CR><LF>
```

### Export von Listenfeldern

Alle Elemente von Listenfeldern werden exportiert, wobei die Nummer des Listenfeldes im Namen des Exportfeldes an den Namen des Listenfeldes angehängt wird (inkl. vorangestelltem Unterstrich).

#### Beispiel:

Das Listenfeld ENTLDIAG hat in MRE\_HE die Exportfelder ENTLDIAG\_1, ENTLDIAG\_2, ...

### Zusatzfelder des Datenexports

Zusatzfelder, welche nicht in der Datenfeldbeschreibung (Tabelle BogenFeld) eines Moduls enthalten sind, werden von der QS-Dokumentationssoftware ausgefüllt.

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.

Erläuterung: 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 oder mit einer kleineren erhält, so wird dieser zurückgewiesen.

- Der Eintrag 1 im Zusatzfeld Storno veranlasst die LAGQH, den übermittelten Datensatz einschließlich seiner Vorversion(en) als „storniert“ zu kennzeichnen.
- Das Zusatzfeld IdBogenFeldMutter wird bei Teildatensätzen eingefügt, welche einen mehrfach anlegbaren Mutterteildatensatz (Attribut fkBogenZahl = '\*' oder '+') haben. In diesem Fall wird die identifizierende Nummer des Mutterteildatensatzes (konfiguriert über Bogen.fkEindeutigBogenFeld) im Kindteildatensatz übermittelt.

Eine vollständige Liste der möglichen Zusatzfelder findet sich in der Tabelle ZusatzFeld der Spezifikationsdatenbank zur QS-Dokumentation.

### **Benennung der Exportdateien**

Exportdateien sollen nach folgendem Schema benannt werden:

M<Modulname><Teildatensatzname>.<Transaktionsnummer>

#### Erläuterung:

- <Modulname> ist der Name des Moduls (Attribut name der Tabelle Modul), wobei der Schrägstrich durch N ersetzt wird
- <Teildatensatzname> ist in Attribut name der Tabelle Bogen zu finden.
- <Transaktionsnummer> : 1, 2, ..., 99, 100, ..., 999, 1000, 1001, ...

#### Beispiele:

MSA\_HEHI\_TIA.001, MMRE\_HEB.12, MSA\_FRUEHREHA\_HEB.13

## **4.1.5. Regeln für die Entgegennahme von Datensätzen und Teildatensätzen**

### **Prüfungen und Datenrückbestätigungen**

Wenn der Datensatz eines Vorgangs aus mehreren Teildatensätzen besteht, so wird der Datensatz des Moduls akzeptiert, wenn alle erforderlichen Teildatensätze fehlerfrei sind. Die entgegennehmende Stelle kann nach der Prüfung eines Datensatzes zu folgenden Ergebnissen (Bestätigungsstatus) kommen:

- OK: Der Datensatz ist fehlerfrei.
- FEHLER: Der Kerndatensatz enthält Fehler und ggf. der optionale Datensatz bzw. ein Storno schlägt fehl.
- STORNO: Der Datensatz (inkl. aller Teildatensätze) wurde von der entgegennehmenden Stelle storniert.

#### Achtung:

Bei jeder Übermittlung vom Krankenhaus an die entgegennehmende Stelle muss der gesamte Datensatz mit allen Teildatensätzen neu übermittelt werden (auch wenn er vorher schon bestätigt worden ist)!

Wurden Datensätze mit OK bestätigt, so muss die Erfassungssoftware auch die Möglichkeit anbieten, eine nachträgliche Aktualisierung zu übermitteln.

### **Definitionen der Fehlerfreiheit von Teildatensätzen und Datensätzen**

- Ein Teildatensatz ist dann fehlerfrei, wenn keines seiner Bogenfelder eine Wertebereichsverletzung aufweist und keine der verletzten Plausibilitätsregeln des Moduls einen Bezug zu einem Bogenfeld des Teildatensatzes hat. Dies gilt auch für teildatensatzübergreifende Regeln.
- Der Kerndatensatz ist dann fehlerfrei, wenn seine Teildatensätze fehlerfrei sind.
- Der optionale Datensatz ist dann fehlerfrei, wenn seine Teildatensätze fehlerfrei sind.

Folgende Prüfungen beziehen sich auf den Exportdatensatz und werden von der LAGQH durchgeführt:

- Prüfung auf Lesbarkeit der übermittelten Transaktionsdateien. Hier erfolgt keine Protokollierung in Form von Protokoll- und Fehlerdateien, sondern der Absender erhält eine entsprechende formlose Nachricht.
- Die formale Prüfung der Steuerdatei umfasst die Korrektheit des Formats.
- Die formale Prüfung der Exportdateien umfasst die Existenz der in der Steuerdatei aufgelisteten Exportdateien und die Korrektheit des Formats.
- Die Vollständigkeits- und Versionsprüfung bezieht sich auf einen übermittelten Datensatz, der durch Registriernummer, Vorgangsnummer und Versionsnummer identifizierbar ist.
- Zulässigkeit der Versionsnummer: Wurde der Datensatz bereits übermittelt, muss die Versionsnummer des neu übermittelten Datensatzes größer als die des bereits gespeicherten Datensatzes sein.
- Teildatensätze mit dem Attribut fkBogenZahl + oder 1 müssen (bezogen auf den Mutterteildatensatz) mindestens einmal oder genau einmal in einem Datensatz existieren.
- Für mehrfach angelegte Teildatensätze mit der Kardinalität + oder \* (Attribut fkBogenZahl Tabelle Bogen) muss die Eindeutigkeit des Feldes fkEindeutigBogenFeld gewährleistet sein.
- Für jeden Teildatensatz mit mehrfach anlegbaren Mutterteildatensätzen ist zu prüfen, ob ein Mutterteildatensatz existiert, für den die Inhalte des Attributs IdBogenFeldMutter und des identifizierenden Attributs des Mutterteildatensatzes (definiert durch fkEindeutigBogenFeld in der Tabelle Bogen) gleich sind.

### **Prüfungen auf doppelte Datensätze**

Ein übermittelter Datensatz wird von der LAGQH nicht angenommen, wenn bereits vorher ein anderer Datensatz mit derselben Registriernummer, Vorgangsnummer und Versionsnummer übermittelt worden ist.

Die Wertebereichsüberprüfungen einschließlich Muss-/Kann-Prüfung und die Plausibilitätsprüfungen werden für jeden übermittelten Datensatz durchgeführt.

### **Prüfung der warnenden Plausibilitätsregeln**

Bei der Datenentgegennahme werden neben den harten Plausibilitätsprüfungen auch die warnenden Plausibilitätsprüfungen (Wert D in Attribut fkRegelTyp der Tabelle Regeln) durchgeführt.

Warnungen werden genauso wie harte Plausibilitätsverletzungen in der Fehlerdatei (inkl. Fehlermeldung, Regelnummer und Regeltyp) an das Krankenhaus übermittelt. Sie sind unterscheidbar anhand des Feldes Regeltyp in der Fehlerdatei.

Warnungen führen nicht zur Ablehnung von Datensätzen durch die LAGQH.

#### Beispiel:

Wenn ein Datensatz keine harte Plausibilitätsprüfung, aber ein oder mehrere warnende Plausibilitätsprüfungen („Warnhinweis“) verletzt, so wird er von der LAGQH angenommen und bestätigt. Für diesen Datensatz werden dem Krankenhaus Warnungen übermittelt, sodass das Krankenhaus die Möglichkeit hat, einen korrigierten Datensatz zu übermitteln.

### **Stornierung von Datensätzen**

Datensätze können storniert werden, damit der Vollständigkeitsabgleich der gelieferten Datensätze

korrekt geführt werden kann.

Dazu ist das Zusatzfeld Storno = 1 zu setzen. Die LAGQH wird dadurch veranlasst, den betreffenden Datensatz einschließlich aller Vorversionen und Teildatensätze als „storniert“ zu kennzeichnen. Der Stornovorgang wird in der Datenbestätigung protokolliert.

Achtung:

Der mit dem Storno-Zusatz gekennzeichnete Datensatz muss ebenfalls eine um eins erhöhte Versionsnummer enthalten, um die Stornierung unabhängig von der Reihenfolge der Verarbeitung von Datensätzen sicherzustellen. Ein Storno mit einer bereits verwendeten Versionsnummer wird zurückgewiesen (Bestätigungsstatus FEHLER, Fehlerart DOPPELT). Ein Stornoversuch eines noch nicht übermittelten Datensatzes wird ebenfalls zurückgewiesen.

#### 4.1.6. Die Antwortdateien

Die Antwortdatei ist ein mit Triple-DES (3DES) verschlüsseltes zip-Archiv und wird wie folgt benannt:

A-<Länder-Kode><Registrierkode>-<Transaktionsnummer>.ZIP.3DES

Beispiel:

A-HE1234A-001.ZIP.3DES

Das unverschlüsselte zip-Archiv enthält die Protokolldatei und die Fehlerdatei, welche von der LAGQH als Antwort auf eine eingegangene Transaktionsdatei generiert werden, und hat einen Namen nach folgendem Schema:

A-<Länder-Kode><Registrierkode>-<Transaktionsnummer>.ZIP

Beispiel:

A-HE1234A-001.ZIP

#### Die Bestätigungsdatei

Die Bestätigungsdatei enthält für jeden übermittelten Datensatz das Ergebnis der Prüfung auf Plausibilität und einen Namen nach folgendem Schema:

B-<Länder-Kode><Registrierkode>.<Transaktionsnummer>

Beispiel:

BHE1234A.001

Der einzelne Datensatz kann über die Kombination von Registriernummer, Vorgangsnummer und Versionsnummer eindeutig identifiziert werden. Im Fehlerfall (Bestätigungsstatus FEHLER) werden die Fehler in einer separaten Datei aufgelistet:

<RegistrierNr>;<VorgangsNr>;<VersionNr>;<Modul>;<SPEZJAHR>;<BestätigungsStatus><CR>  
<LF>

mit:

|                      |   |
|----------------------|---|
| <RegistrierNr>       | Registriernummer des Dokumentationssystems          |
| <VorgangsNr>         | Vorgangsnummer                                      |
| <VersionNr>          | Versionsnummer                                      |
| <Modul>              | Name des übermittelten Moduls (Attribut Modul.name) |
| <SPEZJAHR>           | Jahr der Spezifikation (Tabelle Version)            |
| <BestätigungsStatus> | Werte sind OK, STORNO oder FEHLER                   |
| <CR>                 | Carriage Return (ASCII 13)                          |
| <LF>                 | Line Feed (ASCII 10)                                |

Beispiel:

HE1234A;260612345;1;MRE\_HE;2024;FEHLER<CR><LF>

HE1234A;260612345;1;SA\_HE;2024;OK<CR><LF>

## Die Fehlerdatei

Die Erfassungssoftware muss in der Lage sein, die Fehlermeldungen und Warnungen der LAGQH einzulesen und den Anwender durch eine möglichst komfortable Nachbearbeitungsfunktion für die betreffenden Datensätze zu unterstützen. Die Fehlerdatei hat einen Namen nach folgendem Schema:

F-<Länder-Kode><Registrierkode>.<Transaktionsnummer>

### Beispiel:

FHE1234A.001

Voraussetzung für diese Funktionalität ist, dass von der LAGQH die Fehlermeldungen in einer Fehlerdatei übermittelt werden. Die Zeilen der Fehlerdatei sind folgendermaßen aufgebaut:

<RegistrierNr>;[<Modul>;[<VorgangsNr>;[<VersionNr>;[<SPEZJAHR>;<Fehlerart>;  
[<RegelNr>;[<Regeltyp>;[<Liste>;[<Meldung>]<CR><LF>

mit:

|                |   |
|----------------|---|
| <RegistrierNr> | Registriernummer des Dokumentationssystems  |
| <Modul>        | Modul   |
| <VorgangsNr>   | Vorgangsnummer  |
| <VersionNr>    | Versionsnummer  |
| <SPEZJAHR>     | Jahr der Spezifikation (Tabelle Version)  |
| <Fehlerart>    | STEUER = Formatfehler der Steuerdatei<br>EXPORT = Formatfehler der Exportdatei<br>DOPPELT = bereits vorhandener Datensatz wird erneut übermittelt<br>TDS = Vollständigkeit und Version der Teildatensätze<br>WERT = Wertebereichsverletzung<br>REGEL = Plausibilitätsverletzung |
| <RegelNr>      | Nummer der Regel (idRegeln in Tabelle Regeln),<br>nur wenn Fehlerart = REGEL vergeben, ansonsten leer   |
| <Regeltyp>     | nimmt die Werte H (= hart) oder D (= Warnung)   |
| <Liste>        | In Abhängigkeit von der Fehlerart entweder Liste von Teildatensätzen<br>oder von Bogenfeldern   |
| <Meldung>      | Fehlermeldung als Freitext, z. Z. nicht fest vorgegeben. Für jeden<br>aufgetretenen Fehler wird eine eigene Zeile angehängt!  |
| [ ]            | Option  |
| <CR>           | Carriage Return (ASCII 13)  |
| <LF>           | Line Feed (ASCII 10)  |

Die Fehlerarten beziehen sich auf die in Abschnitt 3.5 beschriebenen Prüfungen. Es gibt für jeden Fehler auszufüllende Spalten bzw. Felder (<RegistrierNr> und <Fehlerart>). Andere Spalten bleiben bei einzelnen Fehlerarten leer, z. B. braucht bei einem Fehler der Steuerdatei nur die Fehlermeldung übermittelt zu werden. Die folgende Tabelle gibt einen Überblick darüber, unter welchen Bedingungen in den Feldern der Fehlerdatei Angaben erforderlich sind.

Ausfüllen der Felder der Fehlerdatei in Abhängigkeit von den Fehlerarten:

| Fehlerart | <Modul> | <VorgangsNr><br><VersionNr> | <SPEZ-<br>VERSION> | <RegelNr> | <Regeltyp> | <Liste>          | <Meldung> |
|-----------|---------|-----------------------------|--------------------|-----------|------------|------------------|-----------|
| STEUER    | -       | -                           | -                  | -         | -          | -                | ja        |
| EXPORT    | ja      | -                           | ja                 | -         | -          | <Bogen>          | ja        |
| DOPPELT   | ja      | ja                          | ja                 | -         | -          | -                | ja        |
| TDS       | ja      | ja                          | ja                 | -         | -          | <Bogenliste>     | ja        |
| WERT      | ja      | ja                          | ja                 | -         | -          | <Bogenfeldliste> | ja        |
| REGEL     | ja      | ja                          | ja                 | ja        | ja         | <Bogenliste>     | ja        |

Die Bogenliste umfasst ein 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 ein 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 (Beispiel: B.AUFNDATUM).

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 RegelFelder 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 das Feld <Regelnummer> 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.). Mit der Nummer des betreffenden Teildatensatzes ist der Inhalt des in der Tabelle BogenFeld unter „fkEindeutigBogenFeld“ definierten Feldes gemeint.

#### Beispiel einer Fehlerdatei:

```
HE1234A;SA_HE;62;1;2024;WERT;;HI_TIA.SCNHSSA;Der Wert '45' des Datenfeldes
SA_HE:HI_TIA:SCNHSSA "NIH-Stroke Scale bei Aufnahme" (Zeile 30) ist größer als '42'<CR><LF>
HE1234A;SA_HE;69;1;2024;REGEL;33;HI_TIA;Gefäßverschluss" ist nicht "ja" [1], obwohl "M2"
ausgefüllt ist<CR><LF>
```

#### **Standardisierung der Meldungen bei Bestätigungsstatus mit Fehlerart DOPPELT:**

Es wurde bereits ein anderer Datensatz mit derselben Registriernummer und Versionsnummer übermittelt.

Standardisierung der Meldungen bei der Fehlerart WERT:

Bei feldbezogenen Fehlern sind die standardisierten Fehlermeldungen aus Abschnitt 3.5.1 zu verwenden.

#### **Standardisierung der Meldungen bei Bestätigungsstatus mit Fehlerart TDS**

Wenn ein obligatorischer Teildatensatz (Attribut Bogen.fkBogenZahl ist + oder 1, oder ein zu einem Kindteildatensatz zugehöriger Mutterteildatensatz) eines Vorgangs in den Exportdateien einer Transaktion nicht vorkommt, so ist folgende Fehlermeldung auszugeben:



Erforderlicher Teildatensatz <Bogen.name> ("**<Bogen.bezeichnung>**") existiert nicht.

Wenn die Existenzbedingung eines Kindteildatensatzes im zugehörigen Mutterteildatensatz erfüllt ist, aber kein Kindteildatensatz vorhanden ist, so ist folgende Fehlermeldung auszugeben (Abschnitt 3.4.2):

Die Angaben im Datensatz erfordern einen Teildatensatz <Bogen.name> ("**<Bogen.bezeichnung>**"). Dieser fehlt.

Wenn die Existenzbedingung eines Kindteildatensatzes im zugehörigen Mutterteildatensatz nicht erfüllt ist, aber trotzdem ein Kindteildatensatz existiert, so ist folgende Fehlermeldung zu erzeugen:

Die Angaben im Datensatz lassen keinen weiteren Teildatensatz <Bogen.name> ("**<Bogen.bezeichnung>**") zu, obwohl ein solcher übermittelt wurde.

Dieser Fehler kann auf zwei Arten hervorgerufen werden:

- Im TDS SA\_HE:B wird ein TDS SA\_HE:HI\_TIA erwartet, aber ein SA\_HE:ICB\_SAB geliefert.
- Es werden zwei TDS SA\_HE:HI\_TIA geliefert.