Datenbankmodell „Virtuelles Fahrzeug“ Version 1.2

Michael Bellair

gesetzt mit TEX/LATEX

Ausgabedatum: 2010-10-11

Zusammenfassung

Das Datenbankmodell „Virtuelles Fahrzeug“ wird beschrieben. Dieses Dokument ist der GNU Free Documentation License (GNU-FDL) unterstellt und somit im Rahmen dieser Lizenz frei veränderbar. Jede veränderte Version ist jedoch selbst unter der GNU-FDL zu veröffentlichen.
Die Lizenz kann Anhang A entnommen werden. Eine deutsche Übersetzung ist unter http://www.giese-online.de/gnufdl-de.html zu finden.

Inhaltsverzeichnis

1 Einleitung

Es wird die Struktur des Datenbankmodells vorgestellt. Jedem Datenbankschema ist ein Abschnitt zugeordnet. Ziel und Zweck des Schemas werden beschrieben.

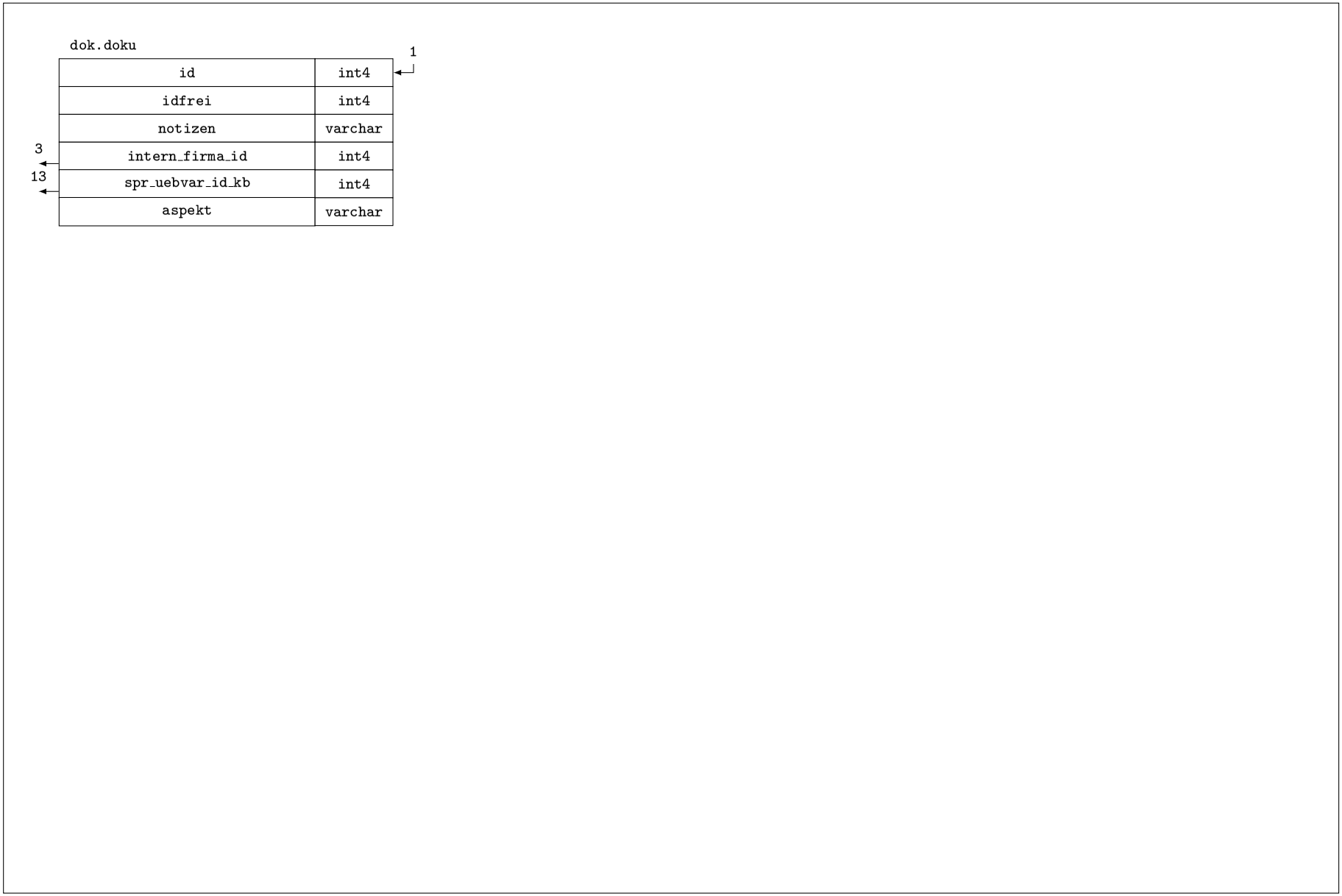
2 Das Datenbankmodell

Datenbank des Virtuellen Fahrzeugs

3 dok

Im Schema „Dokumente“ werden die in der Datenbank verwendeten Dokumentenreferenzen gehalten.

3.1 Tabellenhierarchie der Ebenen 1 — 3



3.2 Schlüssel und Bedingungen

Primärschlüssel

pk__dok_doku_id für Spalte dok.doku.id

Fremdschlüssel

fk__dok_doku__intern_firma_id aus Spalte dok.doku.intern_firma_id

fk__dok_doku__spr_uebvar_id__kb aus Spalte dok.doku.spr_uebvar_id_kb

Unique Constraints

uc__dok_doku in Tabelle dok.doku für die Spalte(n) dok.doku.idfrei, dok.doku.intern_firma_id, dok.doku.spr_uebvar_id_kb

Check Constraints

cc__dok_doku in Tabelle dok.doku die Bedingung (((aspekt)::text = 'BEWEIS'::text) OR ((aspekt)::text = 'QUELLE'::text))

3.3 dok.nimm__doku

Funktionsname

dok.nimm__doku(IN liste_werte varchar[], OUT dokuid varchar)

Funktionsparameter/Rückgabewert(e)

liste_werte Liste der Werte, welche in dok.doku eingelesen werden sollen (Hinweise siehe Funktionsbeschreibung — Werteliste einlesen). Sie werden in der folgenden Reihenfolge in die Spalten

1. spr_uebvar_id_kb
2. intern_firma_id
3. aspekt
4. notizen

eingetragen. Es müssen mindestens die ersten zwei Werte übergeben werden.
Für aspekt darf nur BEWEIS zur Kennzeichnung von Dokumenen, die beweisen, daß eine Sicherheitsanforderung umgesetzt wurde bzw. QUELLE zur Kennzeichnung von Dokumenten, die auf den Ursprung einer Sicherheitsanforderung verweisen.

dokuid Es wird die Datenbank-Identifikationsnummer der eingelesenen Dokumentenreferenz (dok.doku.id) zurückgegeben.

Funktionsbeschreibung

Mittels dieser Funktion werden die im System zu referenzierenden Dokumente in dok.doku eingelesen.

Werteliste einlesen:
Dem Funktionsparameter `liste_werte` ist eine Werteliste (`ARRAY`) zu übergeben (die maximale Anzahl und die Reihenfolge der Werte ist fest vorgegeben — siehe oben). Soll nur eine Teilmenge der Werte übergeben werden, dann sind alle Werte von links nach rechts zu spezifizieren (nichtbenötigte Werte am Ende der Liste dürfen weggelassen werden).

ACHTUNG
Die Beispiele müssen aktualisiert werden !

Beispiele

```
-- Fehlermeldung. Der Parameter liste_werte darf nicht NULL sein.
SELECT * FROM dok.nimm__doku( NULL );

-- Fehlermeldung. Es sind mindestens zwei Werte zu übergeben.
SELECT * FROM dok.nimm__doku( ARRAY['08-15'] );

-- Fehlermeldung. Die Werte müssen größer 0 bzw. dürfen nicht leer sein.
SELECT * FROM dok.nimm__doku( ARRAY['', '1'] );
SELECT * FROM dok.nimm__doku( ARRAY['08-15', '0'] );

-- Dokumentenreferenz einlesen
SELECT dok_doku_id, doknr, intern_firma_id_eigentuemer, spr_uebvar_id__doktitel,
spr_uebvar_id__kurzbeschr, dok_doku_notizen FROM dok.zeige__doku;
SELECT * FROM dok.nimm__doku( ARRAY['08-15', '1'] );
SELECT * FROM dok.nimm__doku( ARRAY['08-15', '1', spr.nimm__wgruppe( 'Dokumententitel',
'1', '0' )] );
SELECT * FROM dok.nimm__doku( ARRAY['08-15', '1', spr.nimm__wgruppe( 'Dokumententitel',
'1', '0' ), spr.nimm__wgruppe( 'Dokumentenkurzbeschreibung', '1', '0' )] );
SELECT * FROM dok.nimm__doku( ARRAY['08-15', '1', spr.nimm__wgruppe( 'Dokumententitel',
'1', '0' ), spr.nimm__wgruppe( 'Dokumentenkurzbeschreibung', '1', '0' ), 'Zusatzinform
);
SELECT dok_doku_id, doknr, intern_firma_id__eigentuemer, spr_uebvar_id__doktitel,
spr_uebvar_id__kurzbeschr, dok_doku_notizen FROM dok.zeige__doku;
```

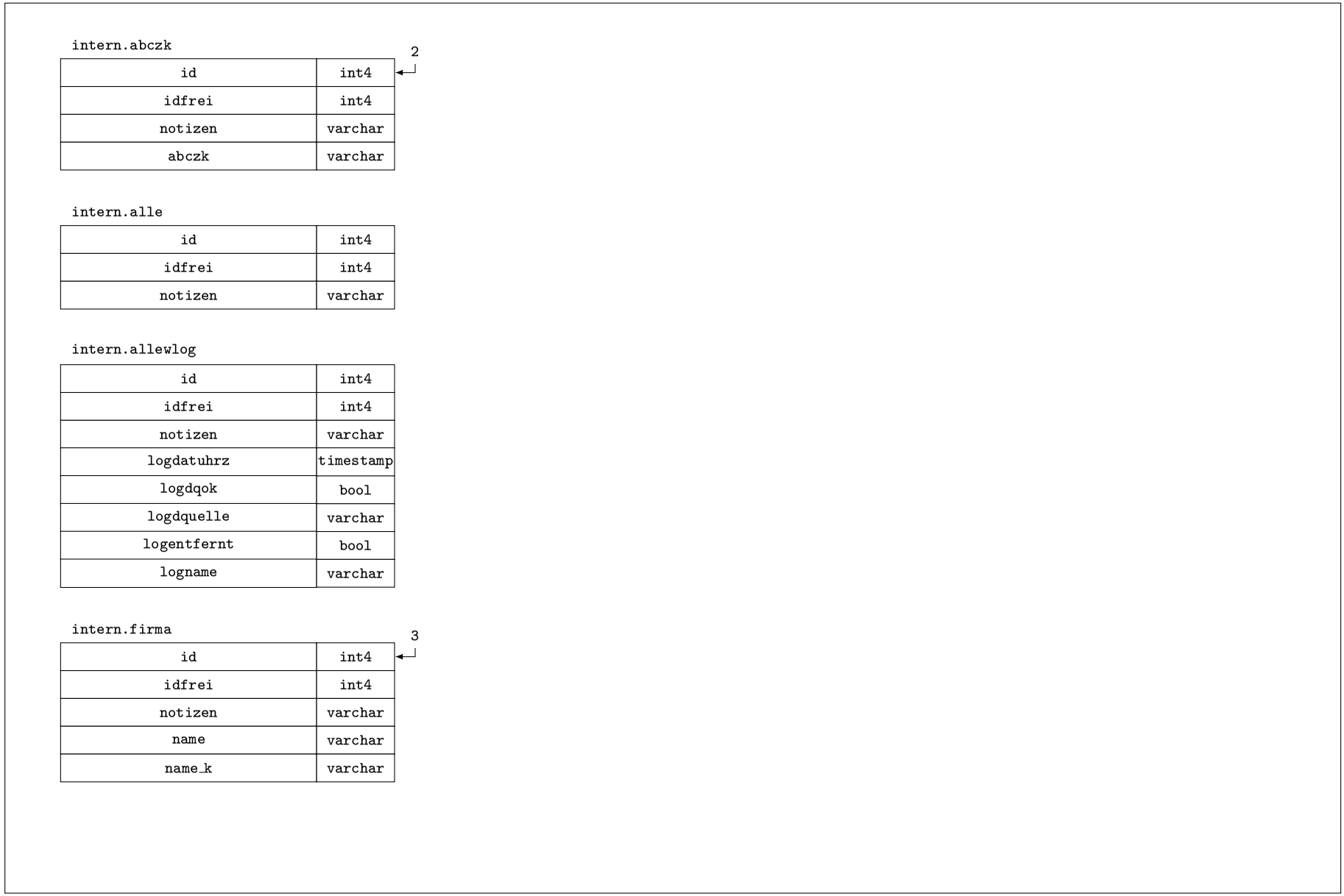
4 intern

Das Schema `Intern` dient der Bereitstellung von Tabellen, Datenstrukturen, usw. welche in allen Schemas verwendet werden können. Weiterhin wird die Basis der Logbuch-Funktionalität einschließlich der Logbuch-Funktion der Tabellen

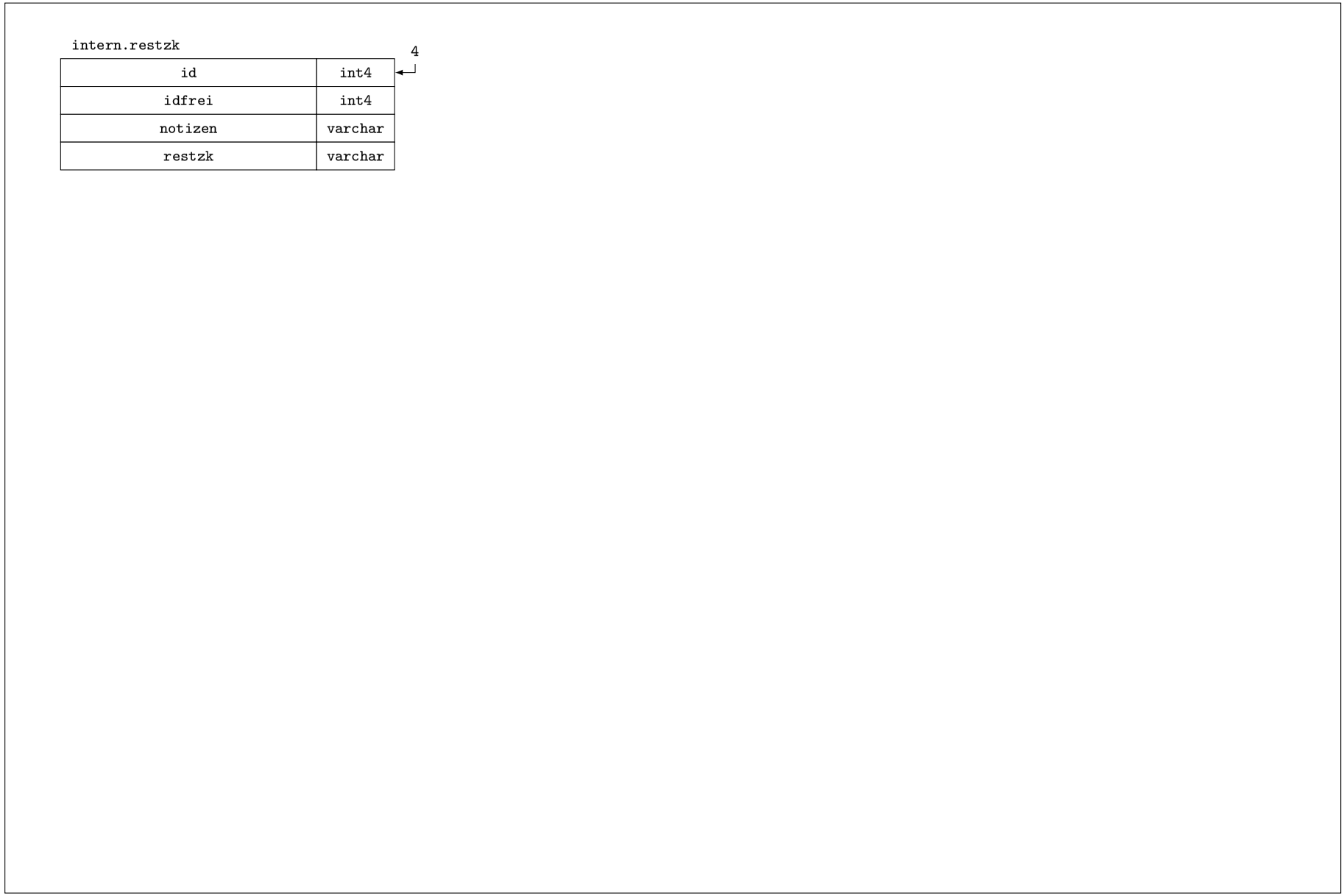
- `abczk`,
- `restzk`

bereitgestellt.

4.1 Tabellenhierarchie der Ebenen 1 — 3



4.2 Tabellenhierarchie der Ebenen 1 — 3



4.3 Schlüssel und Bedingungen

Primärschlüssel

pk__intern_abczk_id für Spalte intern.abczk.id

pk__intern_firma_id für Spalte intern.firma.id

pk__intern_restzk_id für Spalte intern.restzk.id

Unique Constraints

uc__intern_abczk in Tabelle intern.abczk für die Spalte(n) intern.abczk.idfrei, intern.abczk.a

uc__intern_firma in Tabelle intern.firma für die Spalte(n) intern.firma.idfrei, intern.firma.n
intern.firma.name_k

uc__intern_restzk in Tabelle intern.restzk für die Spalte(n) intern.restzk.idfrei,
intern.restzk.restzk

Check Constraints

cc__intern_alle in Tabelle intern.alle die Bedingung ((id = 0) AND (idfrei = 0))

cc__intern_allewlog in Tabelle intern.allewlog die Bedingung (((((((id = 0) AND (idfrei
= 0)) AND (logdatuhrz IS NOT NULL)) AND (logname IS NOT NULL)) AND (logdquelle
IS NULL)) AND (logdqok IS NULL)) AND (logentfernt = false))

4.4 intern.aggregat__zk

Funktionsname
intern.aggregat__zk()

Funktionsparameter/Rückgabewert(e)

-

Funktionsbeschreibung
Diese Aggregatfunktion verbindet sämtliche Teilzeichenketten zu einer Zeichenkette.

4.5 intern.gib__ere_abczk

Funktionsname
intern.gib__ere_abczk(OUT ere_abczk varchar)

Funktionsparameter/Rückgabewert(e)

ere_abczk Ist der erweiterte regulaere Ausdruck zur Beschreibung einer AbcZK.

Funktionsbeschreibung
Ziel der Funktion ist das Definieren und Zurückgeben eines erweiterten regulären Ausdruckes (ERE), welcher das Textmuster eine Zeichenkette beschreibt, die ausschließlich aus Buchstaben besteht - (AbcZK)- (siehe `intern.gib__ERE_Buchstabe(...)`).

Beispiele
```
-- ERE zurückgeben
SELECT * FROM intern.gib__ERE_ABCZK();

-- hallo zurückgeben
SELECT substring( 'hallo' from intern.gib__ERE_ABCZK() );

-- hal zurückgeben
SELECT substring( 'hal2lo' from intern.gib__ERE_ABCZK() );
```

4.6 intern.gib__ere_buchstabe

Funktionsname
`intern.gib__ere_buchstabe( IN istanfang bool, OUT ere_buchstabe varchar )`

Funktionsparameter/Rückgabewert(e)

`istanfang` Der erweiterte reguläre Ausdruck eines Anfangsbuchstaben soll zurückgegeben werden

`ere_buchstabe` Soll der erweiterte reguläre Ausdruck zur Beschreibung eines Anfangsbuchstaben zurückgegeben werden, ist `istAnfang = true` zu setzen. Soll der erweiterte reguläre Ausdruck zur Beschreibung eines Folgebuchstaben zurückgegeben werden, ist `istAnfang = false` zu setzen.

Funktionsbeschreibung
Ziel der Funktion ist das Definieren und Zurueckgeben eines erweiterten regulären Ausdruckes (ERE), welcher das Textmuster eines Anfangsbuchstaben oder Folgebuchstabens beschreibt. Die folgenden Buchstaben werden vom ERE erfaßt:

- A-Z
- a-z
- äöü
- ÄÖÜ
- ß (nur Folgebuchstabe)

Beispiele

```
-- ERE für Anfangsbuchstaben zurückgeben
SELECT * FROM intern.gib__ERE_Buchstabe( true );

-- ERE für die Buchstaben im Wort zurückgeben
SELECT * FROM intern.gib__ERE_Buchstabe( false );

-- h zurückgeben
SELECT substring( 'h' from intern.gib__ERE_Buchstabe( true ) );

-- Leerzeichenkette zurückgeben
SELECT substring( 'ß' from intern.gib__ERE_Buchstabe( true ) );

-- ß zurückgeben
SELECT substring( 'ß' from intern.gib__ERE_Buchstabe( false ) );
```

4.7 intern.gib__ere_restzk

Funktionsname

`intern.gib__ere_restzk( OUT ere_restzk varchar )`

Funktionsparameter/Rückgabewert(e)

`ere_restzk` Ist der erweiterte reguläre Ausdruck zur Beschreibung der Zeichenketten, die keine Buchstaben enthalten (in `intern.restzk` gültig sind).

Funktionsbeschreibung

Ziel der Funktion ist das Definieren und Zurückgeben eines erweiterten regulären Ausdruckes (ERE), welcher das Textmuster der Zeichenkette beschreibt, die über die Tastatur eingebbar ist, aber nicht aus Buchstaben besteht.

Beispiele

```
-- ERE zurückgeben
SELECT * FROM intern.gib__ERE_RestZK();
```

4.8 intern.gib__liste_abczkrestzk

Funktionsname

`intern.gib__liste_abczkrestzk( IN wortgruppe varchar, OUT zk varchar, OUT istabczk bool )`

Funktionsparameter/Rückgabewert(e)

`wortgruppe` Wortgruppe, welche in `abczk` bzw. `restzk` zerlegt werden soll

`zk` Zeichenkette aus der `wortgruppe`, welche eine `abczk` oder `restzk` ist

`istabczk` Wenn `istabczk = true` dann ist die in `zk` vorliegende Zeichenkette eine `abczk` sonst ist sie eine `restzk`

Funktionsbeschreibung
Ziel der Funktion ist das Zerlegen der Wortgruppe `wortgruppe` in `abczk` bzw. `restzk` mittels Erweiterter Regulaerer Ausdruecke (ERE).

Die Vorschrift zum Erkennen einer

- `abczk` kann `intern.gib__ERE_ABCZK()` bzw.
- `restzk` kann `intern.gib__ERE_RestZK()`

entnommen werden.

Beispiele
```
-- Fehlermeldung. Die Wortgruppe ist anzugeben.
SELECT * FROM intern.gib__Liste_abczkrestzk( NULL );
SELECT * FROM intern.gib__Liste_abczkrestzk( '' );

-- Es werden die Bestandteile der Wortgruppe zurückgegeben
SELECT '>' || zk || '<' AS zk, istabczk FROM intern.gib__Liste_abczkrestzk( 'Das
ist ein Array - This is an Array - 1984E' );
```

4.9 intern.gib__liste_tabellenpositionen

Funktionsname
`intern.gib__liste_tabellenpositionen( IN schname varchar, OUT tabname varchar, OUT ebene varchar )`

Funktionsparameter/Rückgabewert(e)

`schname` Name des Schemas dessen Tabellenpositionen (Ebenen innerhalb der Tabellenhierarchie) ermittelt werden.

`tabname` Tabellenname des Schemas

`ebene` Position (Ebene) der Tabelle im Schema

Funktionsbeschreibung
Ziel der Funktion ist das Ermitteln der Positionen (Ebenen) sämtlicher Tabellen im Baum der Tabellenverbindungen (Fremdschlüssel) des Schemas `schName`.

Beispiele
```
-- Fehlermeldung. Der Parameter schName darf nicht NULL oder Leer sein.
```

```
SELECT * FROM intern.gib__Liste_Tabellenpositionen( '' );

-- Tabellenpositionen ermitteln
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'dok' );
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'intern' );
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'kmp' );
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'prj' );
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'prt' );
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'ps' );
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'spr' );
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'str' );
SELECT * FROM intern.gib__Liste_Tabellenpositionen( 'tl' );
```

4.10 intern.gib__naechsteid

Funktionsname
`intern.gib__naechsteid( IN tabname varchar, OUT tabnameid varchar )`

Funktionsparameter/Rückgabewert(e)

`tabname` Name der Tabelle (ggf. einschliesslich Schemaname), aus der die nächste verfügbare Identifikationsnummer `id` ermittelt werden soll. (darf nicht leer sein).

`tabnameid` Nächste freie `id`

Funktionsbeschreibung
Ziel der Funktion ist das Ermitteln der nächsten verfügbaren `id` der übergebenen Tabelle `tabName`. Es wird der kleinste mögliche Wert gesucht.

Beispiele
```
-- Fehlermeldung. Der Parameter tabName darf nicht NULL sein.
SELECT intern.gib__naechsteid( NULL );

-- Eine wiederverwendbare id wurde nicht gefunden
SELECT max(id) FROM intern.abczk;
SELECT intern.gib__naechsteid( 'intern.abczk' );

-- Eine wiederverwendbare id wurde gefunden
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWort']
);
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWortZwe
);
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;
SELECT intern.loesche__zeile( 'intern.abczk', 'abczk = ''KeinRichtigesWort''' );
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;
```

```
SELECT intern.gib__naechsteid( 'intern.abczk' );
```

4.11 intern.gib__sqlselect

Funktionsname
`intern.gib__sqlselect( IN sel varchar, IN sel_from varchar[], IN sel_where varchar, OUT sqlselect varchar )`

Funktionsparameter/Rückgabewert(e)

`sel` Es ist die Liste der Parameter anzugeben, welche das Ergebnis der `SELECT` - Abfrage sind. Diese Variable darf nicht leer sein.

`sel_from` Es ist die Liste (`ARRAY`) der Tabellen anzugeben, welche mittels `SELECT` - Abfrage ausgewertet werden sollen. Diese Variable darf nicht leer sein.

`sel_where` Es ist eine `WHERE` - Bedingung anzugeben (ggf. einschließlich Semikolon). Diese Variable darf leer sein.

`sqlselect` Es wird die `SELECT` - Anweisung zurueckgegeben.

Funktionsbeschreibung
Ziel der Funktion ist das Erzeugen einer `SELECT` - Anweisung, welche innerhalb einer `PL/pgSQL` - Funktion verwendet werden kann. Sie berücksichtigt das Konzept zur Wiederverwendung gelöschter `id`. Das abschliessende Semikolon ist explizit anzugeben.

Beispiele
```
-- Fehlermeldung. Der Parameter sel darf nicht leer oder NULL sein.
SELECT intern.gib__sqlSELECT( '', '', '' );

-- Fehlermeldung. Der Parameter sel_from darf nicht leer oder NULL sein.
SELECT intern.gib__sqlSELECT( 'hsk', NULL, '' );

-- Fehlermeldung. Kein Element der Liste sel_from darf leer sein
SELECT intern.gib__sqlSELECT( 'hsk', ARRAY[''], '' );

-- SQL-Abfrage erzeugen
SELECT intern.gib__sqlSELECT( 'id, idfrei, notizen', 'intern.alle', '' );
SELECT intern.gib__sqlSELECT( 'id, idfrei, notizen', 'tt.muell, tt.test', ' AND
tt.muell.id = tt.test.tt_muell_id' );
```

4.12 intern.gib__stsname

Funktionsname
`intern.gib__stsname( IN befehl varchar, IN vollstname varchar, OUT name varchar`

)

Funktionsparameter/Rückgabewert(e)

`befehl` In Form eines fest vorgegebenen Befehls ist anzugeben, welcher Teil des vollständigen Spaltennamens zurückzugeben ist:

sch Schemaname

tab Tabellenname

sp Spaltenname

schtab Schemaname und Tabellenname

tabsp Tabellenname und Spaltenname

Die Gültigkeit der übergebenen Befehle wird nicht überprüft. Es darf nicht `NULL` bzw. keine Leerzeichenkette angegeben werden.

`vollstname` Es ist der vollständige Spaltenname anzugeben. Es wird geprüft, ob die Syntax eingehalten wurde.

`name` Es wird der Name zurückgebene, welcher in `befehl` spezifiziert wurde.

Funktionsbeschreibung
Ziel der Funktion ist das Extrahieren des Schema,- Tabellen- oder Spaltennamens aus dem vollständigen Spaltennamen. Der vollständige Spaltenname besteht aus dem Schemanamen, dem Tabellennamen und dem Spaltennamen welche mittels Punkt getrennt sind und aus Buchstaben `a-z` bzw. `A-Z`, Ziffern `0-9` bzw. Unterstrich bestehen.
Im Fehlerfall wird eine Leerzeichenkette zurückgegeben.

Beispiele

```
-- Fehlermeldung. Es ist ein vollständiger Spaltenname anzugeben
SELECT * FROM intern.gib__stsname( 'sp', NULL );
SELECT * FROM intern.gib__stsname( 'sp', '' );

-- Fehlermeldung. Es ist Befehl anzugeben
SELECT * FROM intern.gib__stsname( NULL, 'intern.alle.notizen' );
SELECT * FROM intern.gib__stsname( '', 'intern.alle.notizen' );

-- Fehlermeldung. Der vollständige Spaltenname ist in der Syntax schema.tabelle.spalte
anzugeben
SELECT * FROM intern.gib__stsname( 'sch', 'i.a' );
SELECT * FROM intern.gib__stsname( 'sch', '.a.n' );
SELECT * FROM intern.gib__stsname( 'sch', 'i..n' );
SELECT * FROM intern.gib__stsname( 'sch', 'i.a.' );
SELECT * FROM intern.gib__stsname( 'sch', '..' );
SELECT * FROM intern.gib__stsname( 'sch', 'iß.a.n' );
SELECT * FROM intern.gib__stsname( 'sch', 'i.a2.n' );
SELECT * FROM intern.gib__stsname( 'sch', 'i.a.n,' );
```

```
SELECT * FROM intern.gib__stsname( 'sch', 'i.a.n.falsch' );

-- Fehlermeldung. Der übergebene befehl ist falsch
SELECT * FROM intern.gib__stsname( 'unbekannterBefehl', 'intern.alle.notizen' );

-- Schema, Tabelle, Spalte, ... zurückgeben
SELECT * FROM intern.gib__stsname( 'sch', 'intern.alle.notizen' );
SELECT * FROM intern.gib__stsname( 'tab', 'intern.alle.notizen' );
SELECT * FROM intern.gib__stsname( 'sp', 'intern.alle.notizen' );
SELECT * FROM intern.gib__stsname( 'schtab', 'intern.alle.notizen' );
SELECT * FROM intern.gib__stsname( 'tabsp', 'intern.alle.notizen' );
SELECT * FROM intern.gib__stsname( 'tabsp', 'int2ern.al_le.notizen85' );
```

4.13 intern.gib__zu

Funktionsname
`intern.gib__zu( OUT zu bpchar )`

Funktionsparameter/Rückgabewert(e)

`zu` Das Ersatzzeichen für den Zeilenumbruch

Funktionsbeschreibung
Ziel der Funktion ist das Zurückgeben des im System zugelassenen Ersatzzeichens für den Zeilenumbruch.
Alle Texte, welche ein Zeilenumbruchzeichen enthalten, sollen mit dem hier zurückgegebenen Zeichen erstellt (gespeichert werden). Das Bearbeiten der Datenbankdumps oder der Texte, welche aus der Datenbank ausgelesen werden, wird vereinfacht.

Beispiele
```
-- Es wird das Ersatzzeichen für den Zeilenumbruch zurückgegeben
SELECT * FROM intern.gib__ZU();
```

4.14 intern.ist__abczk

Funktionsname
`intern.ist__abczk( IN abczk varchar, OUT istabczk bool )`

Funktionsparameter/Rückgabewert(e)

`abczk` Es ist die zu testende Zeichenkette anzugeben.

`istabczk` Wenn eine AbcZK vorliegt wird `true` zurückgegeben, sonst `false`.

Funktionsbeschreibung
Ziel der Funktion ist das Prüfen, ob die übergebene Zeichenkette `abczk` als Zeichenkette interpretiert werden kann, welche ausschließlich aus Buchstaben besteht (AbcZK). Die gültigen Buchstaben können `intern.gib__ERE_ABCZK()` entnommen werden.

Beispiele
```
-- Es wird false zurückgegeben
SELECT * FROM intern.ist__abczk( '' );
SELECT * FROM intern.ist__abczk( 'ein-/auslesen' );
SELECT * FROM intern.ist__abczk( 'ß' );
SELECT * FROM intern.ist__abczk( 'ßbeginnt' );

-- Es wird true zurückgegeben
SELECT * FROM intern.ist__abczk( 'I' );
SELECT * FROM intern.ist__abczk( 'daß' );
```

4.15 intern.loesche__zeile

Funktionsname
`intern.loesche__zeile( IN tabname varchar, IN sql_where varchar, IN beachteri varchar )`

Funktionsparameter/Rückgabewert(e)

`tabname` Name einschliesslich Schemaname der Tabelle, aus der die Zeile gelöscht werden soll (darf nicht leer sein)

`sql_where` SQL-Bedingung welche die zu löschende(n) Zeile(n) beschreibt (darf leer sein)

`beachteri` Es ist `true` anzugeben, wenn die referentielle Integrität beachtet werden soll. Sonst ist `false` anzugeben.

Funktionsbeschreibung
Ziel der Funktion ist das Löschen von Zeilen (außer der Zeile mit `id = 0`) unter Berücksichtigung der Bedingungen, welche mittels `sql_where` spezifiziert sind. Wenn `sql_where` leer ist, werden alle Zeilen gelöscht.
Es ist besonders zu beachten, daß `tabName` den Schemanamen enthalten muss !

Beispiele
```
-- Fehlermeldung. Der Tabellenname ist anzugeben
SELECT intern.loesche__zeile( NULL, '' );

-- Fehlermeldung. Die übergebene Tabelle ist nicht vorhanden
SELECT intern.loesche__zeile( 'intern.abc', '' );
```

```
-- Fehlermeldung. Die Zeilen können nicht gelöscht werden, da weitere Zeilen abhängen.
SELECT intern.loesche__zeile( 'intern.abczk', '' );

-- Fehlermeldung. Die übergebe WHERE - Bedingung ist ungültig
SELECT intern.loesche__zeile( 'intern.abczk', 'intern.abczk.id = 99999999' );

-- Tabelleninhalt anzeigen
SELECT intern.nimm__zeile( 'tl.datei', ARRAY['dname'], ARRAY['Dateiname-1']);
SELECT intern.nimm__zeile( 'tl.datei', ARRAY['dname'], ARRAY['Dateiname-2']);
SELECT * FROM tl.datei WHERE id < 5;

-- Gesamten Inhalt der Tabelle löschen
SELECT intern.loesche__zeile( 'tl.datei', NULL );

-- Zeile mit der id = 1 löschen
SELECT intern.loesche__zeile( 'tl.datei', 'tl.datei.id = 1' );

-- Tabelleninhalt anzeigen
SELECT * FROM tl.datei WHERE id < 5;
```

4.16 intern.nimm__tlkommentar

Funktionsname

`intern.nimm__tlkommentar( IN sqldbobj varchar, IN kommentar varchar )`

Funktionsparameter/Rückgabewert(e)

`sqldbobj` Es ist das Datenbankobjekt zu benennen (SQL) welchem der Kommentar hinzugefügt werden soll.

`kommentar` Der Kommentar (einschl. TL-Befehlen). '\' sind zu maskieren !

Funktionsbeschreibung

Ziel der Funktion ist das Hinzufügen eines Kommentares zu einem Datenbankobjekt. Im Kommentar können TL-Befehle verwendet werden, da die Dokumentation zur Datenbank mittels TEX/LATEX formatiert wird.
Zeilenumbrüche werden mittels `intern.gib__ZU()` ersetzt.

Beispiele

```
-- Fehlermeldung. Der Parameter sqlDBObj darf nicht leer oder NULL sein.
SELECT intern.nimm__TLKommentar( '', '' );
SELECT intern.nimm__TLKommentar( NULL, '' );

-- Kommantar gelöscht
SELECT intern.nimm__TLKommentar( 'DATABASE ' || current_database(), '' );
```

```
-- Kommantar hinzugefügt
SELECT intern.nimm__TLKommentar( 'DATABASE ' || current_database(), 'Neuer Kommentar'
);

-- prüfen, ob Kommantar tatsächlich hinzugefügt wurde
SELECT shobj_description( ( SELECT oid FROM pg_database WHERE pg_database.datname
= current_database() LIMIT 1 ), 'pg_database');
```

4.17 intern.nimm__tlkommentarfunktion

Funktionsname

`intern.nimm__tlkommentarfunktion( IN fkt varchar, IN fktparam varchar[], IN beschr varchar, IN beispiel varchar[] )`

Funktionsparameter/Rückgabewert(e)

`fkt` Es ist der Name der Funktion anzugeben, der der Kommentar zugeordnet werden soll. Er darf nicht leer oder NULL sein.

`fktparam` In Form eines ARRAY sind in der Reihenfolge ihres Auftretens sämtliche Beschreibungen der Funktionsparameter anzugeben. Es wird nicht zwischen Eingabeparametern `IN` bzw. Rückgabeparametern `OUT / INOUT` unterschieden. Sollte die Funktion über keine Parameter verfügen, ist `NULL` anzugeben.

`beschr` Es ist die Beschreibung der Funktion einzugeben. Sie darf nicht leer oder `NULL` sein.

`beispiel` Es können mehrere Beispiele zur Demonstration der Nutzung der Funktion angegeben werden. Sie dürfen `NULL` sein.

Funktionsbeschreibung

Ziel der Funktion ist das Hinzufügen einer Beschreibung (eines Kommentars) zu einer Funktion. Hinweise zur Gültigkeit können `intern.nimm__TLKommentar(...)` entnommen werden.

Beispiele

```
-- Fehlermeldung. Die Parameter fkt und beschr dürfen nicht NULL sein.
SELECT intern.nimm__TLKommentarFunktion( NULL, NULL, '', NULL );
SELECT intern.nimm__TLKommentarFunktion( 'intern.nimm__TLKommentarFunktion', NULL,
'', NULL );

-- Kommentar einlesen
SELECT intern.nimm__TLKommentarFunktion( 'intern.nimm__TLKommentarFunktion( fkt
varchar, fktparam varchar[], beschr varchar, beispiel varchar[] )', NULL, 'Testbeschre
NULL );
\df+ intern.nimm__tlkommentarfunktion
```

4.18 intern.nimm__tlkommentarsicht

Funktionsname
`intern.nimm__tlkommentarsicht( IN sicht varchar, IN spnamen varchar[], IN spbeschr varchar[], IN beschr varchar )`

Funktionsparameter/Rückgabewert(e)

`sicht` Es ist der vollständige Name der Sicht anzugeben. Er darf nicht leer oder NULL sein.

`spnamen` In Form eines `ARRAY` sind in der Reihenfolge ihres Auftretens sämtliche Namen der Spalten anzugeben, welche von der Sicht bereitgestellt werden. Es muß mindestens eine Spalte beschrieben werden.

`spbeschr` In Form eines `ARRAY` sind in der Reihenfolge ihres Auftretens sämtliche Beschreibungen der Spalten anzugeben, welche von der Sicht bereitgestellt werden. Es muß mindestens eine Spalte beschrieben werden.

`beschr` Es ist die Sicht zu beschreiben. Es muß eine Beschreibung angegeben werden.

Funktionsbeschreibung
Ziel der Funktion ist das Hinzufügen einer Beschreibung (eines Kommentars) zu einer Sicht. Hinweise zur Gültigkeit können `intern.nimm__TLKommentar(...)` entnommen werden.

Beispiele
```
-- Fehlermeldung. Der Name der Sicht fehlt
SELECT intern.nimm__TLKommentarSicht( NULL, NULL, NULL, '' );

-- Fehlermeldung. Der Kommentar fehlt
SELECT intern.nimm__TLKommentarSicht( 'intern.zeige__fremdschluessel', ARRAY['spName'],
ARRAY['spBeschreibung'], '' );

-- Fehlermeldung. Es ist mindestens eine Spalte zu beschreiben
SELECT intern.nimm__TLKommentarSicht( 'intern.zeige__fremdschluessel', NULL, ARRAY['spBesch
'Kommentar' );
SELECT intern.nimm__TLKommentarSicht( 'intern.zeige__fremdschluessel', ARRAY['spName'],
NULL, 'Kommentar' );

-- Fehlermeldung. Die Anzahl der Spaltennamen muß mit der Anzahl der Spaltenbeschreibungen
übereinstimmen
SELECT intern.nimm__TLKommentarSicht( 'intern.zeige__fremdschluessel', ARRAY['spName1',
'spName2'], ARRAY['spBeschreibung'], 'Kommentar' );

-- Sicht kommentieren, Ergebnis überprüfen
SELECT intern.nimm__TLKommentarSicht( 'intern.zeige__fremdschluessel', ARRAY['SpName'],
ARRAY['SpBeschreibung'], 'Beschreibung' );
\dv+ intern.zeige__fremdschluessel
```

4.19 intern.nimm__tlkommentarsts

Funktionsname
`intern.nimm__tlkommentarsts( IN sts varchar, IN kommentar varchar )`

Funktionsparameter/Rückgabewert(e)

`sts` Es ist der vollständige Schemaname, Tabelnname oder Spaltenname anzugeben.

`kommentar` Es ist der Kommentar anzugeben

Funktionsbeschreibung
Ziel der Funktion ist das Hinzufügen eines Kommentares zu einem Schema, einer Tabelle oder einer Spalte. Es ist der vollständige Name anzugeben (Bsp.: `schema.tabelle.spalte`). Anhand der Anzahl der Datenbankobjekte wird entschieden, ob ein Schema, eine Tabelle oder eine Spalte vorliegt.
Hinweise zur Gültigkeit können `intern.nimm__TLKommentar(...)` entnommen werden.

Beispiele
```
-- Fehlermeldung. Der Parameter sts darf nicht NULL sein.
SELECT intern.nimm__TLKommentarSTS( '', '' );
SELECT intern.nimm__TLKommentarSTS( NULL, '' );

-- Kommantar gelöscht
SELECT intern.nimm__TLKommentarSTS( 'intern.alle.id', '' );

-- Kommantar hinzugefügt
SELECT intern.nimm__TLKommentarSTS( 'intern.alle.id', 'Neuer Kommentar' );
\d+ intern.alle
```

4.20 intern.nimm__zeile

Funktionsname
`intern.nimm__zeile( IN tabname varchar, IN liste_spname varchar[], IN liste_werte varchar[], OUT tabnameid varchar )`

Funktionsparameter/Rückgabewert(e)

`tabname` Name der Tabelle (ggf. einschliesslich Schemaname), in die die Werte `liste_werte[]` eingelesen werden sollen (darf nicht leer sein).

`liste_spname` Liste der Spalten, welche die Werte aufnehmen sollen. Es muß mindestens ein Spaltenname übergeben werden. Es dürfen keine Leerzeichenketten übergeben werden.

`liste_werte` Liste der Werte, welche eingefügt werden sollen. Es muß mindestens ein Wert übergeben werden. Es dürfen keine Leerzeichenketten übergeben werden. Weiterhin muß dieselbe Anzahl Werte übergeben werden, wie Spalten benannt sind.

`tabnameid id` der Zeile, welche die übergebenen Werte aufgenommen hat

Funktionsbeschreibung
Ziel der Funktion ist das Einlesen von Werten. Der Spaltenliste `liste_spName[]` darf jede in der Tabelle `tabName` vorhandene Spalte übergeben werden. Die Spalte `idfrei` darf nicht pbergeben werden, da sie ausschließlich vom System gesetzt wird.
Die übergebenen Werte `liste_werte[]` werden in der DB gesucht und die `id` der Zeile zurückgegeben, welche die Werte bereits enthält. Sollte keine Zeile gefunden werden, wird die nächste verfügbare `id` ermittelt und die Werte eingelesen.

Sollte `liste_spName[]` die Spalte `id` explizit benannt sein, so wird unabhängig vom Wert der Spalte `idfrei` die übergebene `id` gesucht, der gesamte Inhalt gelöscht und die neuen Werte eingetragen. Wurde `id` nicht gefunden, wird eine Zeile mit der uebergebenen `id` hinzugefügt.

Beispiele

```
-- Fehlermeldung. Der Parameter tabName darf nicht NULL oder leer sein.
SELECT intern.nimm__zeile( NULL, ARRAY[''], ARRAY[''] );

-- Fehlermeldung. Die Parameter liste_spName und liste_werte müssen mindestens
ein Element enthalten.
SELECT intern.nimm__zeile( 'intern.abczk', NULL, NULL );
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], NULL );

-- Fehlermeldung. Die Parameter liste_spName und liste_werte dürfen keine Leerzeichenkette
enthalten.
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY[''], ARRAY[''] );
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY[''] );

-- Fehlermeldung. Der Parameter liste_wert muß dieselbe Anzahl Elemente enthalten,
wie der Parameter liste_spName.
SELECT intern.nimm__zeile( 'intern.abczk', 'notizen, abczk', 'zeichenkette für
notizen' );

-- Fehlermeldung. Die Spalte idfrei darf nicht explizit gesetzt werden
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['idfrei'], ARRAY['0'] );

-- Der Wert war bereits in der Datenbank enthalten
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWort']
);
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWort']
);
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;

-- Wiederverwendung einer gelöschten id
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWort']
);
```

```
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWortZwe
);
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;
SELECT intern.loesche__zeile( 'intern.abczk', 'abczk = ''KeinRichtigesWort''' );
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWortEin
);
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;

-- Überschreiben einer Zeile mit vorgegebener id
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWort']
);
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['abczk'], ARRAY['KeinRichtigesWortZwe
);
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['id', 'abczk'], ARRAY['2399',
'KeinRichtigesWortEins'] );
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;

-- Einfügen einer Zeile mit vorgegebener id
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;
SELECT intern.nimm__zeile( 'intern.abczk', ARRAY['id', 'abczk'], ARRAY['999999',
'NichtVorhandeneId'] );
SELECT * FROM intern.abczk ORDER BY id DESC LIMIT 5;
```

4.21 intern.pruefe__array

Funktionsname

`intern.pruefe__array( IN liste varchar[], IN minel int4, IN maxel int4, IN fktname varchar, IN paramname varchar )`

Funktionsparameter/Rückgabewert(e)

`liste` Es ist die Liste anzugeben, deren Plausibiliät geprüft werden soll.

`minel` Es ist die Anzahl der Listenelemente anzugeben, welche mindestens übergeben werden sollen. Wenn kleiner `1` oder `NULL` übergeben wird, muß die Liste `NULL` sein.

`maxel` Es ist die Anzahl der Listenelemente anzugeben, welche maximal übergeben werden sollen. Wenn kleiner `1` oder `NULL` übergeben wird, darf die Liste eine beliebige Anzahl Elemente besitzen. Andernfalls darf die Anzahl der Elemente den übergebenen Wert nicht überschreiten.

`fktname` Es ist der Name der Funktion anzugeben, welche die Prüfroutine aufruft. Der Funktionsname wird in der Fehlermeldung verwendet. Sollte er leer oder `NULL` sein, wird `intern.pruefe__array(...)` verwendet.

`paramname` Es ist der Name des Parameters anzugeben, welcher die zu prüfende Liste `liste` beinhaltet. Der Parametername wird in der Fehlermeldung verwendet. Sollte er leer oder `NULL` sein, wird `liste` verwendet.

Funktionsbeschreibung
Ziel der Funktion ist die Plausibilitätsprüfung der mittels des Parameters `liste` übergebenen Liste (`ARRAY`) anhand der Randbedingungen `minel` und `maxel`. Wenn die Liste nicht `NULL` sein darf, darf keines der Listenelemente leer sein. Sollte die Plausibilitätsprüfung negativ ausfallen, wird eine Fehlermeldung ausgegeben.

Beispiele

```
Fehlermeldung mit Standardfunktions- und Parameternamen ausgeben, wenn liste nicht
NULL sein darf.
SELECT * FROM intern.pruefe__array( NULL, 1, 1, NULL, NULL );
SELECT * FROM intern.pruefe__array( NULL, 1, 1, '', '' );

Fehlermeldung mit übergebenen Funktions- und Parameternamen ausgeben, wenn liste
nicht NULL sein darf.
SELECT * FROM intern.pruefe__array( NULL, 1, 1, 'fktname(...)', 'listenName' );

Fehlermeldung mit Standardfunktions- und Parameternamen ausgeben, wenn liste NULL
sein muß.
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', 'zwei' ], NULL, 1, NULL, NULL
);
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', 'zwei' ], 0, 1, '', '' );
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', 'zwei' ], -5, 1, NULL, NULL
);

Fehlermeldung mit übergebenen Funktions- und Parameternamen ausgeben, wenn liste
NULL sein muß.
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', 'zwei' ], NULL, 1, 'fktname(...)',
'listenName' );

Fehlermeldung ausgeben, wenn liste mindesten 2 Elemente enthalten muß.
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins' ], 2, 0, NULL, NULL );
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins' ], 2, NULL, 'fktname(...)', 'listenName'
);

Fehlermeldung ausgeben, wenn liste maximal 1 Element enthalten darf.
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', 'zwei' ], 1, 1, '', '' );
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', 'zwei' ], 1, 1, 'fktname(...)',
'listenName' );

Fehlermeldung ausgeben, wenn liste maximal 1 Element enthalten darf.
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', '', 'drei' ], 1, 3, NULL, NULL
);
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', '', 'drei' ], 1, 3, '', '' );
```

```
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', '', 'drei' ], 1, 3, 'fktname(...)',
'listenName' );

keine Fehlermeldung ausgeben, weil liste NULL sein darf.
SELECT * FROM intern.pruefe__array( NULL, NULL, 1, 'fktname(...)', 'listenName'
);
SELECT * FROM intern.pruefe__array( NULL, 0, 1, 'fktname(...)', 'listenName' );
SELECT * FROM intern.pruefe__array( NULL, -5, 1, 'fktname(...)', 'listenName' );

keine Fehlermeldung ausgeben, weil liste minimal 1 und maximal 2 Element enthalten
darf.
SELECT * FROM intern.pruefe__array( ARRAY[ 'eins', 'zwei' ], 1, 2, 'fktname(...)',
'listenName' );
```

4.22 intern.pruefe__dt

Funktionsname
`intern.pruefe__dt( IN dt varchar, IN wert varchar, IN ug varchar, IN og varchar, IN fktname varchar, IN paramname varchar )`

Funktionsparameter/Rückgabewert(e)

`dt` Es ist einer der möglichen Datentypen (siehe Funktionsbeschreibung — Gültige Datentypen / Wertebereich) anzugeben

`wert` Es ist der Wert anzugeben, dessen Plausibiliät geprüft werden soll. Er darf nicht `NULL` oder leer sein.

`ug` Es ist die untere Grenze des Wertes anzugeben (weitere Hinweise siehe Funktionsbeschreibung — Gültige Datentypen / Wertebereich). Wird `NULL` oder eine Leerzeichenkette angegeben, gilt die in PostgreSQL implementierte untere Grenze.

`og` Es ist die obere Grenze des Wertes anzugeben. Wird `NULL` oder eine Leerzeichenkette angegeben, gilt die in PostgreSQL implementierte obere Grenze.

`fktname` Es ist der Name der Funktion anzugeben, welche die Prüfroutine aufruft. Der Funktionsname wird in der Fehlermeldung verwendet. Sollte er leer oder `NULL` sein, wird `intern.pruefe__dt(...)` verwendet.

`paramname` Es ist der Name des Parameters anzugeben, welcher den zu prüfenden Wert `wert` beinhaltet. Der Parametername wird in der Fehlermeldung verwendet. Sollte er leer oder `NULL` sein, wird `wert` verwendet.

Funktionsbeschreibung
Ziel der Funktion ist die Plausibilitätsprüfung elementarer Datentypen. Die Datentypen und der Wertebereich werden als Zeichenketten übergeben. Sollte die Plausibilitätsprüfung negativ

ausfallen, wird eine Fehlermeldung ausgegeben.
Der im PostgreSQL implementierte Wertebereich der Datentypen kann der Postgres-Dokumentation entnommen werden. Er wird von PostgreSQL automatisch und somit innerhalb dieser Rputine nicht explizit überprüft.

Gültige Datentypen / Wertebereich
Es werden die folgenden Datentypen geprüft:

bw Boolescher Wert. Die untere / obere Grenze wird nicht überprüft.

du Datum / Uhrzeit. Die untere / obere Grenze wird überprüft.

gz Ganzzahl (integer). Die untere / obere Grenze wird überprüft.

zk Zeichenkette. Die untere Grenze wird nicht überprüft. Die obere Grenze wird überprüft.

Beispiele

```
Fehlermeldung. Der übergebene Datentyp darf nicht NULL oder leer sein
SELECT * FROM intern.pruefe__dt( NULL, '1', NULL, NULL, NULL, NULL );

Fehlermeldung. Der übergebene Datentyp ist unbekannt
SELECT * FROM intern.pruefe__dt( 'unbekannt', '1', NULL, NULL, 'fktname(...)',
'wertName' );

Fehlermeldung: Es ist kein boolean
SELECT * FROM intern.pruefe__dt( 'bw', 'keinBool', 'ignoriert', 'ignoriert', NULL,
NULL );
SELECT * FROM intern.pruefe__dt( 'bw', 'keinBool', 'ignoriert', 'ignoriert', 'fktname(...)
'wertName' );

Fehlermeldung. Der übergebene Wert darf nicht NULL oder leer sein
SELECT * FROM intern.pruefe__dt( NULL, NULL, NULL, NULL, NULL, NULL );
SELECT * FROM intern.pruefe__dt( NULL, '', NULL, NULL, 'fktname(...)', 'wertName'
);

Fehlermeldung. Das Datum befindet sich außerhalb der zulässigen Grenzen
SELECT * FROM intern.pruefe__dt( 'du', '2007-04-16', '2007-04-17', NULL, 'fktname(...)',
'datUhrz' );
SELECT * FROM intern.pruefe__dt( 'du', '2007-04-18', NULL, '2007-04-17', 'fktname(...)',
'datUhrz' );
SELECT * FROM intern.pruefe__dt( 'du', '2007-04-19', '2007-04-17', '2007-04-18',
'fktname(...)', 'datUhrz' );

Fehlermeldung. Die Ganzzahl (integer) befindet sich außerhalb der zulässigen Grenzen
SELECT * FROM intern.pruefe__dt( 'gz', '-2', '-1', NULL, 'fktname(...)', 'gz_int'
);
```

```
SELECT * FROM intern.pruefe_dt( 'gz', '2', NULL, '1', 'fktname(...)', 'gz_int'
);
SELECT * FROM intern.pruefe_dt( 'gz', '-5', '-1', '1', 'fktname(...)', 'gz_int'
);

Fehlermeldung. Die übergebene Zeichenkette ist zu lang
SELECT * FROM intern.pruefe_dt( 'zk', '1234567890', NULL, '9', NULL, NULL );
SELECT * FROM intern.pruefe_dt( 'zk', '1234567890', NULL, '9', 'fktname(...)',
'zkWert' );

keine Fehlermeldung
SELECT * FROM intern.pruefe_dt( 'bw', 'TRUE', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', 'T', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', 'Y', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', 'YES', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', '1', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', 'FALSE', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', 'F', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', 'N', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', 'NO', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'bw', '0', NULL, NULL, 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'du', '2007-04-18', NULL, NULL, 'fktname(...)',
'wertName' );
SELECT * FROM intern.pruefe_dt( 'du', '2007-04-18', '2007-04-17', '2007-04-19',
'fktname(...)', 'wertName' );
SELECT * FROM intern.pruefe_dt( 'gz', '2123456', NULL, NULL, 'fktname(...)', 'wertNam
);
SELECT * FROM intern.pruefe_dt( 'gz', '0', '-1', '1', 'fktname(...)', 'wertName'
);
SELECT * FROM intern.pruefe_dt( 'zk', '123456789', 'ignoriert', '9', 'fktname(...)',
'wertName' );
```

4.23 intern.verbindezk

Funktionsname
`intern.verbindezk( IN zk varchar, IN zkadd varchar, OUT zkges varchar )`

Funktionsparameter/Rückgabewert(e)

`zk` Erste Zeichenkette.

`zkadd` Zweite Zeichenkette.

`zkges` Erste + Zweite Zeichenkette.

Funktionsbeschreibung
Ziel der Funktion ist das Verbinden beider Zeichenketten. Diese Funktion wird in der Aggregatfunktion `intern.erzeuge_zk(...)` verwendet.

Beispiele
```
intern.verbindezk( 'Anfang ', '- Ende' );
```

4.24 intern.zeige_fremdschluessel

Beschreibung der Spalten

`fk` Namen der Fremdschlussel in der Datenbank

`spstart` Schema.Tabelle.Name der Startspalte des Fremdschlüssels

`spziel` Schema.Tabelle.Name der Zielspalte des Fremdschlüssels

Beschreibung der Sicht
Es werden alle im System vorhandenen Fremdschlüssel einschließlich der Start- und Zielspalten aufgelistet.

4.25 intern.zeige_spaltenanzahl

Beschreibung der Spalten

`tabname` Tabellenname

`spanz` Anzahl der Spalten der Tabelle

Beschreibung der Sicht
Es wird die Anzahl der in der Tabelle enthaltenen Spalten (ohne Systemspalten) aufgelistet.

4.26 intern.zeige__tabspinformationen

Beschreibung der Spalten

`spname` Schema.Tabelle.Name der Spalte

`dtyp` pg_type.typname

`spnr` pg_attribute.attnum

`beschr_spalte` pg_description.description für die Spalte

`beschr_tabelle` pg_description.description für die Tabelle

Beschreibung der Sicht

Es werden alle im System vorhandenen Spalten (einschließlich Schema- und Tabellennamen) und deren Beschreibung aufgelistet.

5 prj

Ausschließlich im Schema „Projekte“ wird der Projektbezug für alle weiteren Schemas hergestellt. Die Definition des Begriffes „Projekt“ hängt selbstverständlich von der Anwendung ab.

In der vorliegenden Datenbank kann mittels des Projektes „DBDokumentation“ die Dokumentation der Datenbank aus den Kommentaren der verschiedenen Datenbankobjekte erzeugt werden.
Dazu sind in der Kommandozeile (Eingabeaufforderung engl. Shell) die folgenden Befehle einzugeben:

1. `psql -U micha vf -P tuples_only -P format=unaligned -P footer -c ‘‘SELECT replace( makefile, ’-U micha vf’, ’-U ’ || current_user || ’ ’ || current_database() ) FROM prj.projekte WHERE prjname = ’DBDokumentation’;‘‘ -o makefile`
   Anstelle des Benutzers `micha` im shell-Befehl `psql -U micha vf ...` und des Datenbanknamens `vf` ist der eigene Benutzername bzw. Datenbankname anzugeben

2. `make`

Das genannte Verfahren kann für sämtliche Projekte, denen ein `makefile` zugeordnet ist, angewendet werden. Mögliche Varianten zum Erzeugen unterschiedlicher Dokumente können dem `makefile` des entsprechenden Projektes entnommen werden.

Benennen der Tabellen

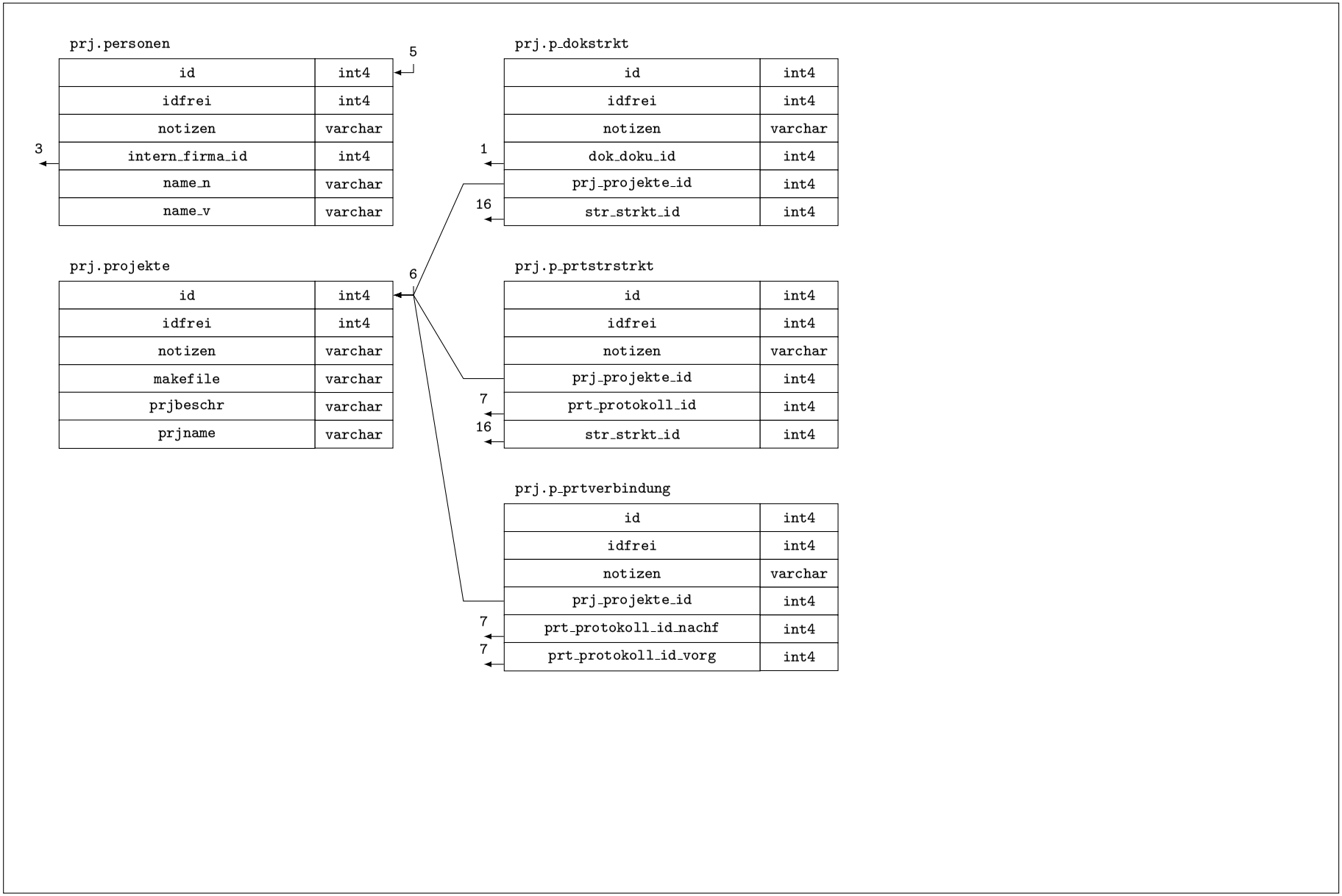
`prj.p_*` Projektbezug herstellen

`prj.pp_*` projektbezogene Personenzuordnung

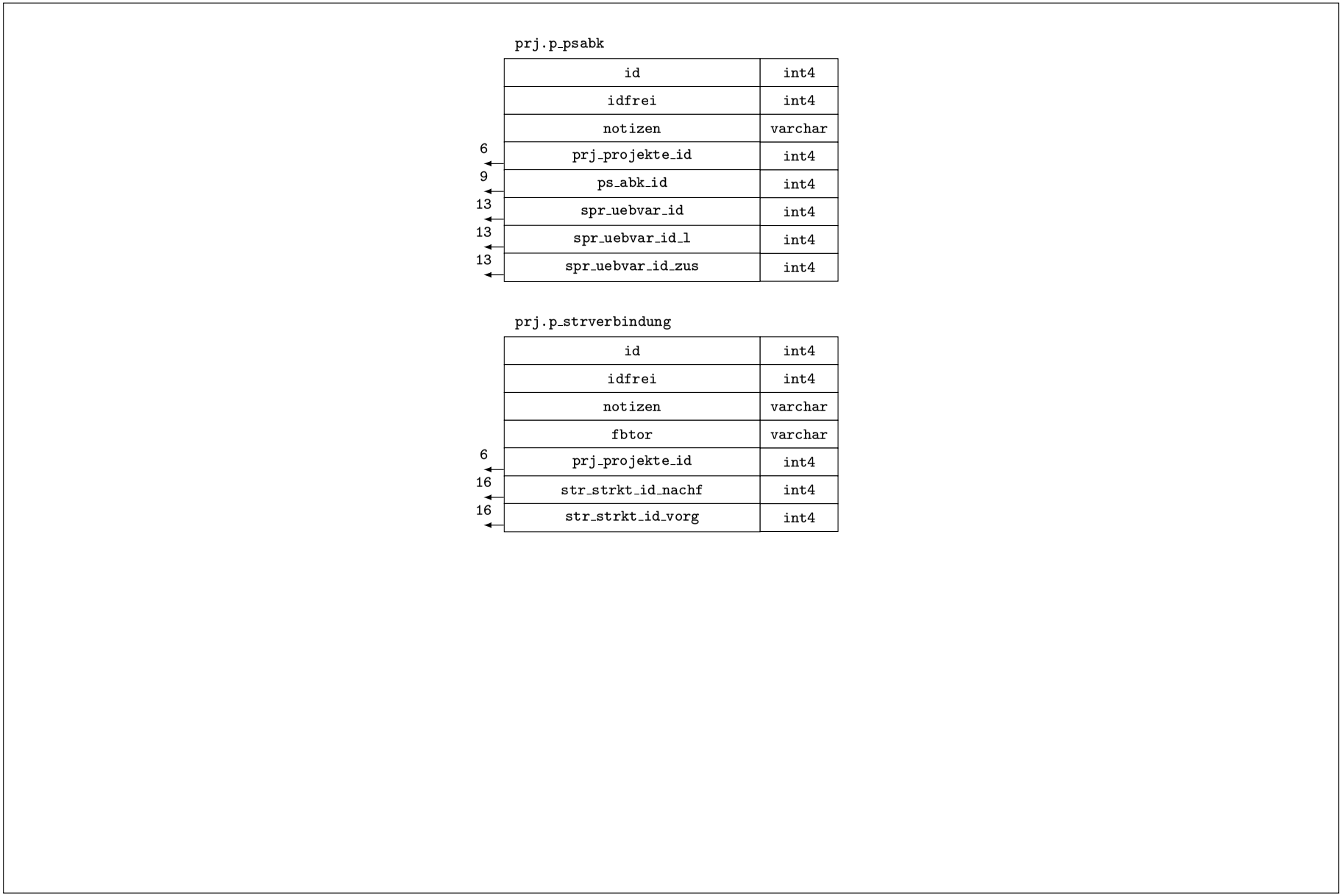
`prj.pl_*` projektbezogene Logbuchfunktionalität

`prj.pplu_*` projektbezogene Logbuchfunktionalität und Personenzuordnung

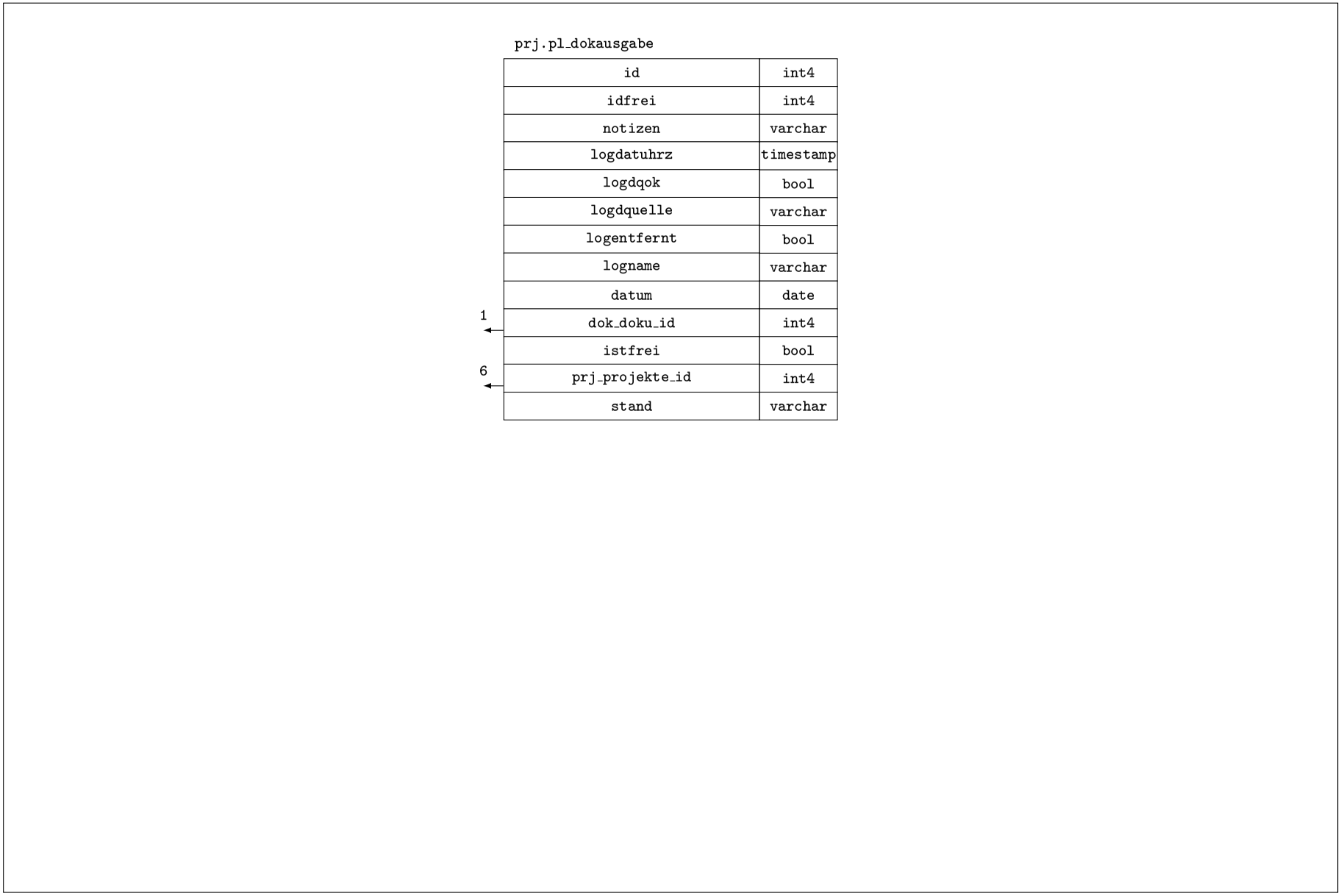
5.1 Tabellenhierarchie der Ebenen 1 — 3



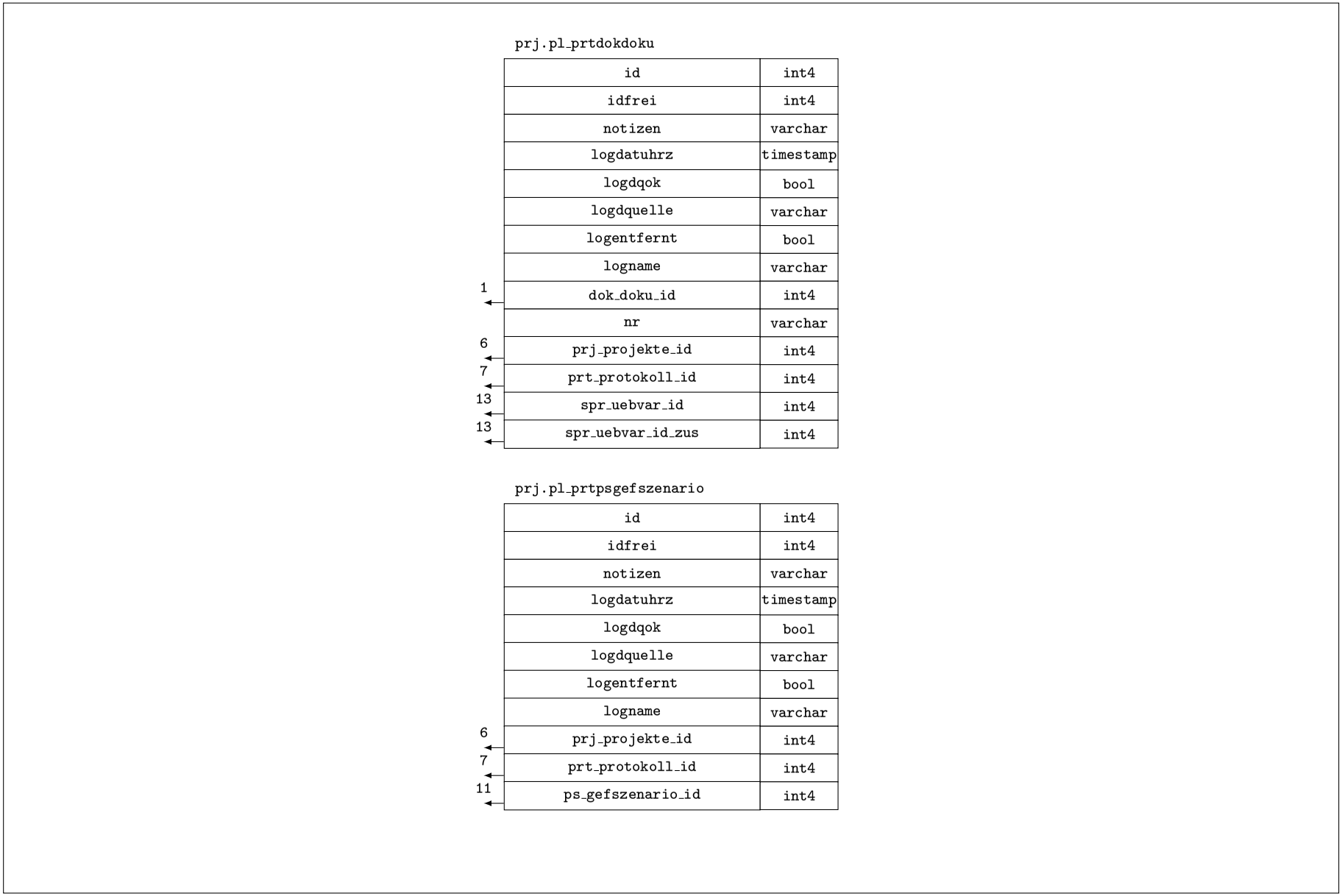
5.2 Tabellenhierarchie der Ebenen 1 — 3



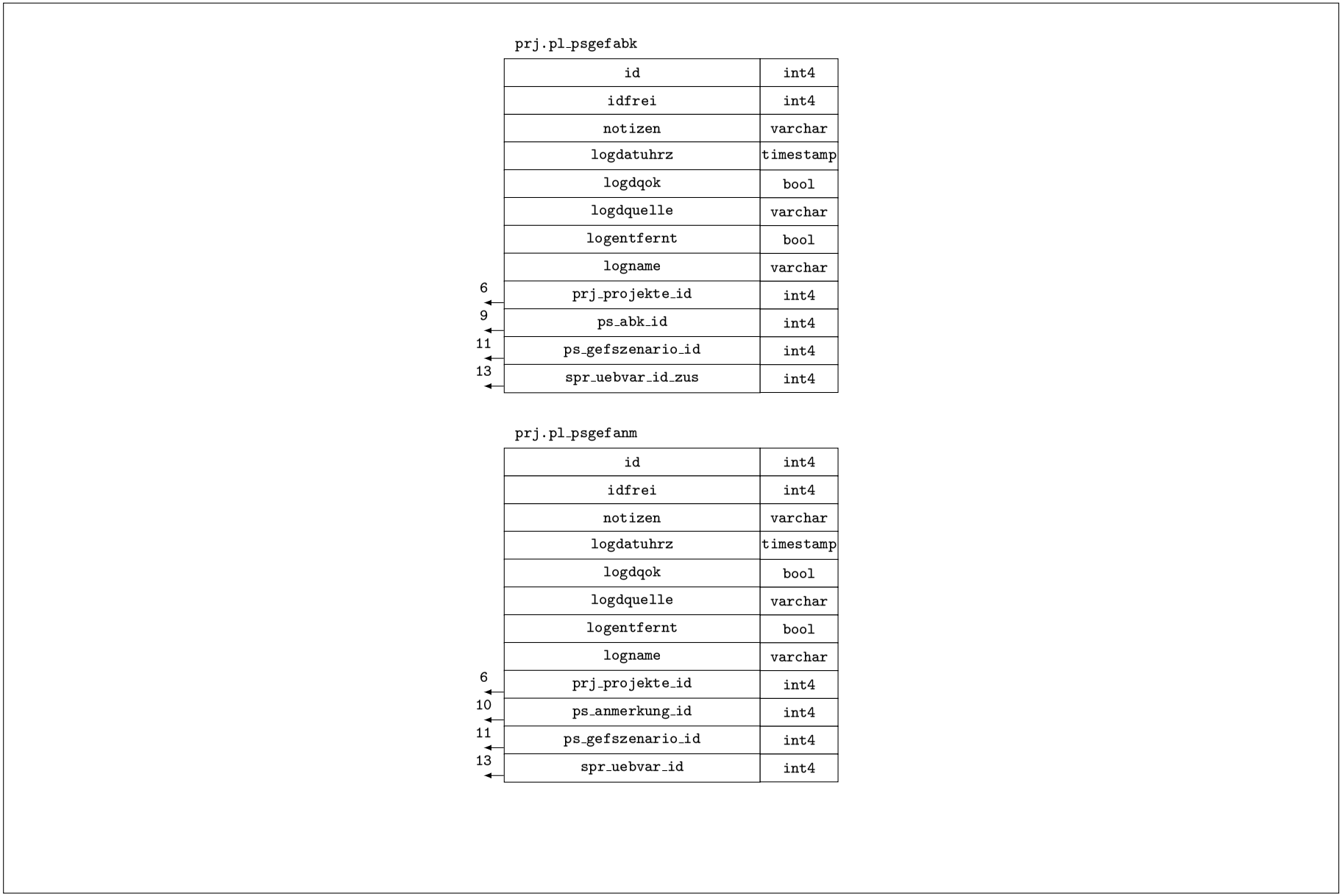
5.3 Tabellenhierarchie der Ebenen 1 — 3



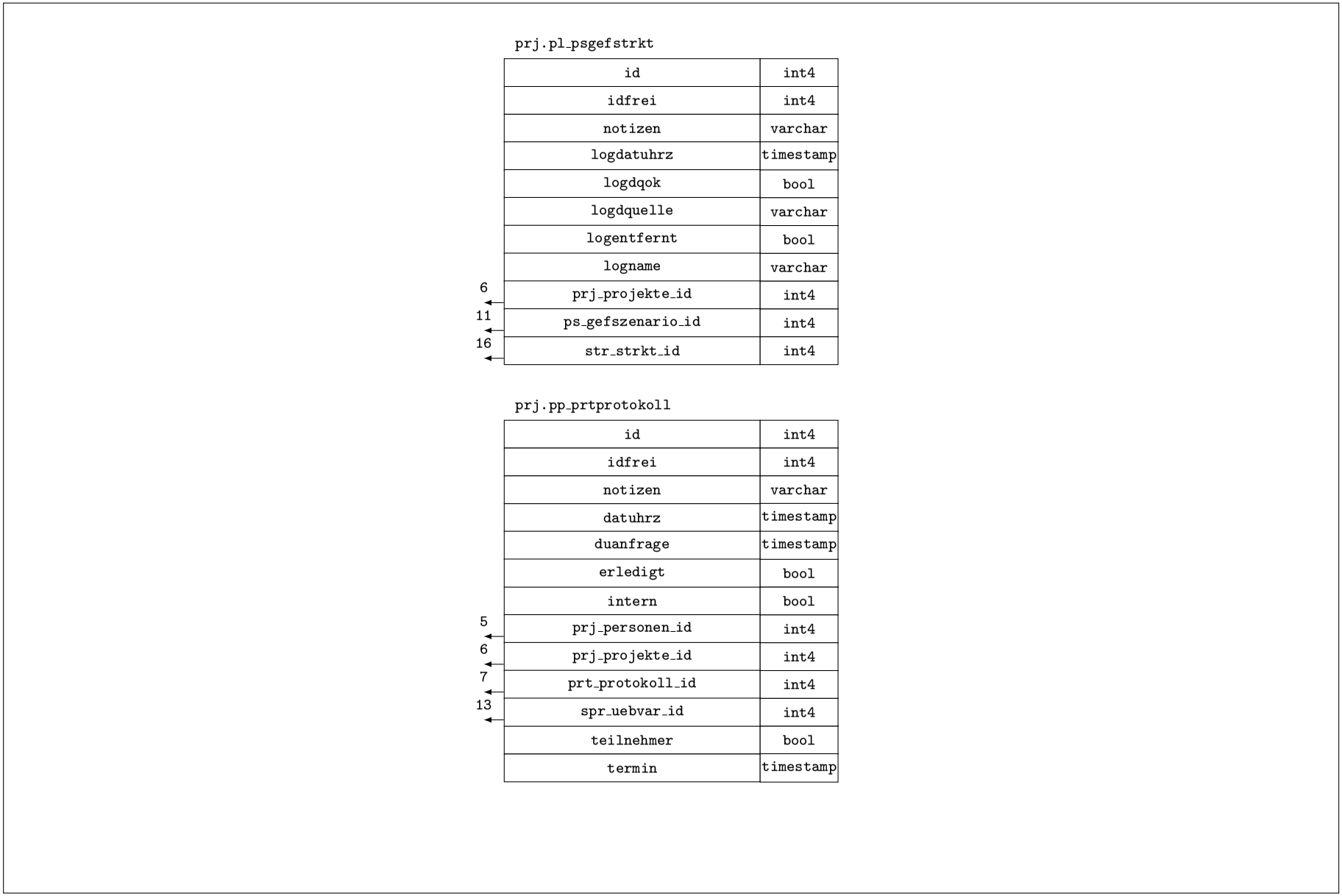
5.4 Tabellenhierarchie der Ebenen 1 — 3



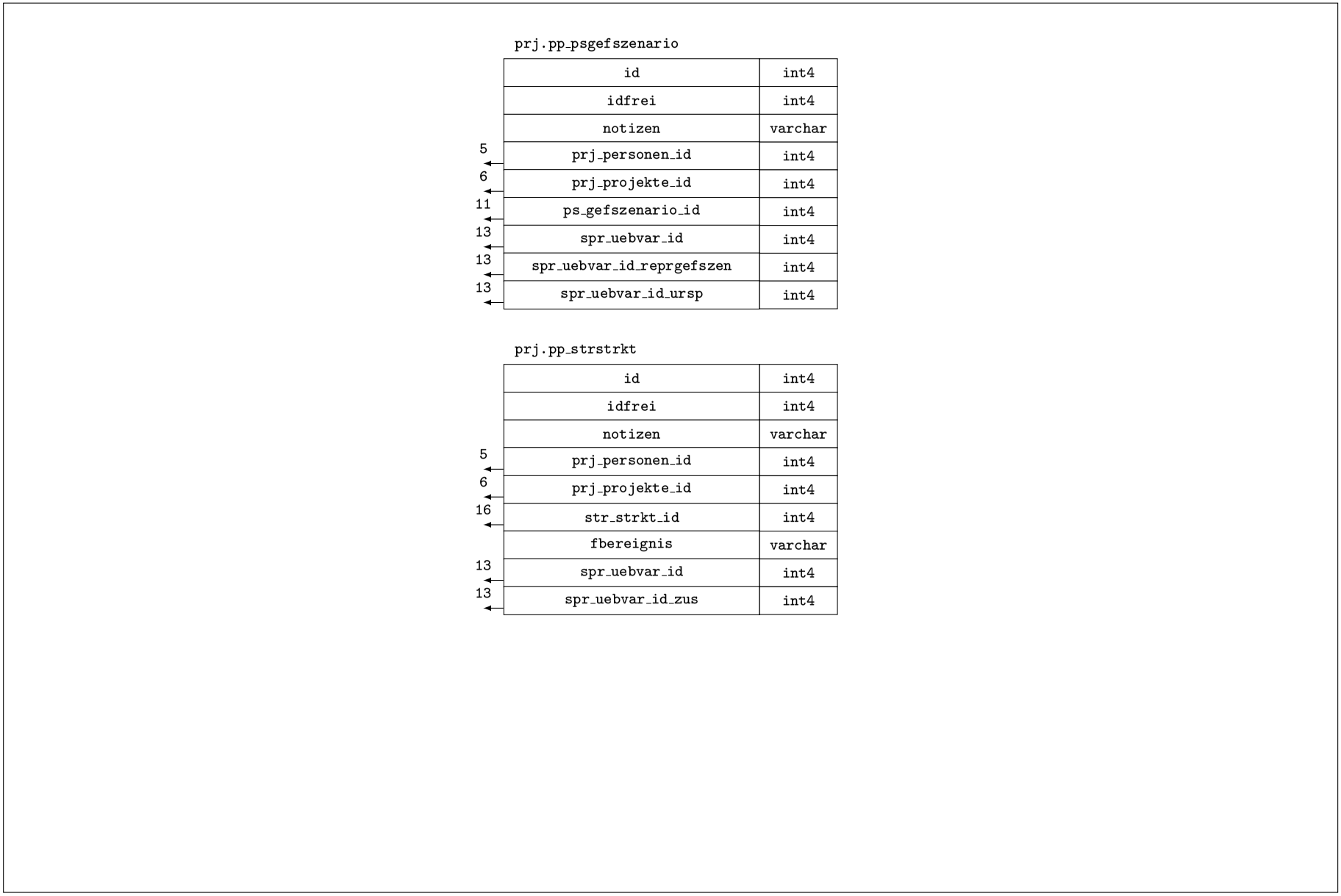
5.5 Tabellenhierarchie der Ebenen 1 — 3



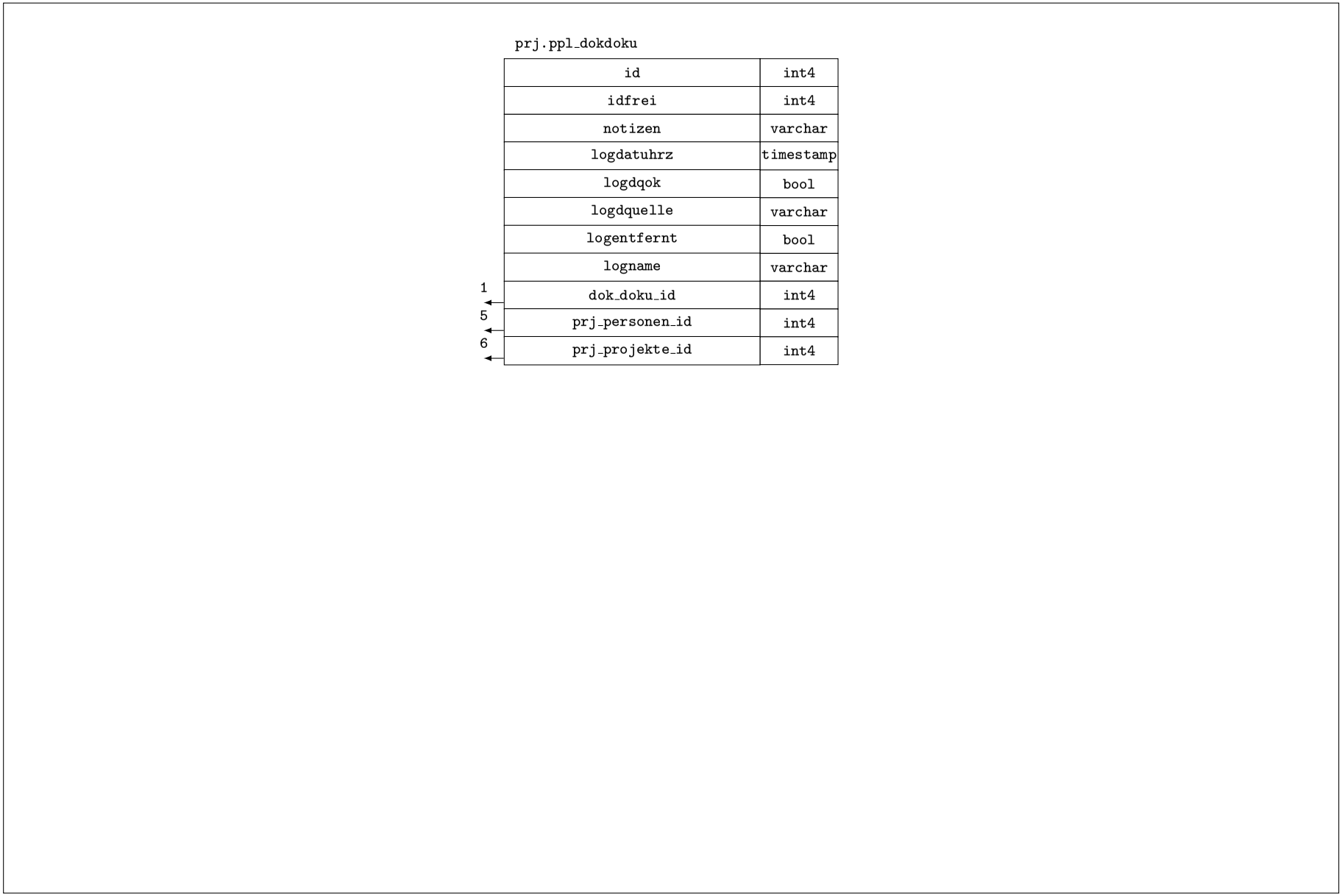
5.6 Tabellenhierarchie der Ebenen 1 — 3



5.7 Tabellenhierarchie der Ebenen 1 — 3



5.8 Tabellenhierarchie der Ebenen 1 — 3



5.9 Schlüssel und Bedingungen

Primärschlüssel

pk__prj_p_dokstrkt_id für Spalte prj.p_dokstrkt.id

pk__prj_p_prtstrstrkt_id für Spalte prj.p_prtstrstrkt.id

pk__prj_p_prtverbindung_id für Spalte prj.p_prtverbindung.id

pk__prj_p_psabk_id für Spalte prj.p_psabk.id

pk__prj_p_strverbindung_id für Spalte prj.p_strverbindung.id

pk__prj_personen_id für Spalte prj.personen.id

pk__prj_pl_dokausgabe_id für Spalte prj.pl_dokausgabe.id

pk__prj_pl_prtdokdoku_id für Spalte prj.pl_prtdokdoku.id

pk__prj_pl_prtpsgefszenario_id für Spalte prj.pl_prtpsgefszenario.id

pk__prj_pl_psgefabk_id für Spalte prj.pl_psgefabk.id

pk__prj_pl_psgefstrkt_id für Spalte prj.pl_psgefstrkt.id

pk__prj_pp_protokoll_id für Spalte prj.pp_prtprotokoll.id

pk__prj_pp_psgefszenario_id für Spalte prj.pp_psgefszenario.id

pk__prj_pp_strstrkt_id für Spalte prj.pp_strstrkt.id

pk__prj_ppl_dokdoku_id für Spalte prj.ppl_dokdoku.id

pk__prj_projekte_id für Spalte prj.projekte.id

Fremdschlüssel

fk__prj_p_dokstrkt__dok_doku_id aus Spalte prj.p_dokstrkt.dok_doku_id

fk__prj_p_dokstrkt__prj_projekte_id aus Spalte prj.p_dokstrkt.prj_projekte_id

fk__prj_p_dokstrkt__str_strkt_id aus Spalte prj.p_dokstrkt.str_strkt_id

fk__prj_p_prtstrstrkt__prj_projekte_id aus Spalte prj.p_prtstrstrkt.prj_projekte_id

fk__prj_p_prtstrstrkt__prt_protokoll_id aus Spalte prj.p_prtstrstrkt.prt_protokoll_id

fk__prj_p_prtstrstrkt__str_strkt_id aus Spalte prj.p_prtstrstrkt.str_strkt_id

fk__prj_p_prtverbindung__prj_projekte_id aus Spalte prj.p_prtverbindung.prj_projekte_id

fk__prj_p_prtverbindung__prt_protokoll_id_nachf aus Spalte prj.p_prtverbindung.prt_protokol

fk__prj_p_prtverbindung__prt_protokoll_id_vorg aus Spalte prj.p_prtverbindung.prt_protokoll

fk__prj_p_psabk__prj_projekte_id aus Spalte prj.p_psabk.prj_projekte_id

fk__prj_p_psabk__ps_abk_id aus Spalte prj.p_psabk.ps_abk_id

fk__prj_p_psabk__spr_uebvar_id aus Spalte prj.p_psabk.spr_uebvar_id

fk__prj_p_psabk__spr_uebvar_id_l aus Spalte prj.p_psabk.spr_uebvar_id_l

fk__prj_p_psabk__spr_uebvar_id_zus aus Spalte prj.p_psabk.spr_uebvar_id_zus

fk__prj_p_strverbindung__prj_projekte_id aus Spalte prj.p_strverbindung.prj_projekte_id

fk__prj_p_strverbindung__str_strkt_id_nachf aus Spalte prj.p_strverbindung.str_strkt_id_nachf

fk__prj_p_strverbindung__str_strkt_id_vorg aus Spalte prj.p_strverbindung.str_strkt_id_vorg

fk__prj_personen__intern_firma_id aus Spalte prj.personen.intern_firma_id

fk__prj_pl_dokausgabe__dok_doku_id aus Spalte prj.pl_dokausgabe.dok_doku_id

fk__prj_pl_dokausgabe__prj_projekte_id aus Spalte prj.pl_dokausgabe.prj_projekte_id

fk__prj_pl_prtdokdoku__dok_doku_id aus Spalte prj.pl_prtdokdoku.dok_doku_id

fk__prj_pl_prtdokdoku__prj_projekte_id aus Spalte prj.pl_prtdokdoku.prj_projekte_id

fk__prj_pl_prtdokdoku__prt_protokoll_id aus Spalte prj.pl_prtdokdoku.prt_protokoll_id

fk__prj_pl_prtdokdoku__spr_uebvar_id aus Spalte prj.pl_prtdokdoku.spr_uebvar_id

fk__prj_pl_prtdokdoku__spr_uebvar_id_zus aus Spalte prj.pl_prtdokdoku.spr_uebvar_id_zus

fk__prj_pl_prtpsgefszenario__prj_projekte_id aus Spalte prj.pl_prtpsgefszenario.prj_projekte_id

fk__prj_pl_prtpsgefszenario__prt_protokoll_id aus Spalte prj.pl_prtpsgefszenario.prt_protokoll_

fk__prj_pl_prtpsgefszenario__ps_gefszenario_id aus Spalte prj.pl_prtpsgefszenario.ps_gefszenari

fk__prj_pl_psgefabk__prj_projekte_id aus Spalte prj.pl_psgefabk.prj_projekte_id

fk__prj_pl_psgefabk__ps_abk_id aus Spalte prj.pl_psgefabk.ps_abk_id

fk__prj_pl_psgefabk__ps_gefszenario_id aus Spalte prj.pl_psgefabk.ps_gefszenario_id

fk__prj_pl_psgefabk__spr_uebvar_id_zus aus Spalte prj.pl_psgefabk.spr_uebvar_id_zus

fk__prj_pl_psgefanm__prj_projekte_id aus Spalte prj.pl_psgefanm.prj_projekte_id

fk__prj_pl_psgefanm__ps_anmerkung_id aus Spalte prj.pl_psgefanm.ps_anmerkung_id

fk__prj_pl_psgefanm__ps_gefszenario_id aus Spalte prj.pl_psgefanm.ps_gefszenario_id

fk__prj_pl_psgefanm__spr_uebvar_id aus Spalte prj.pl_psgefanm.spr_uebvar_id

fk__prj_pl_psgefstrkt__prj_projekte_id aus Spalte prj.pl_psgefstrkt.prj_projekte_id

fk__prj_pl_psgefstrkt__ps_gefszenario_id aus Spalte prj.pl_psgefstrkt.ps_gefszenario_id

fk__prj_pl_psgefstrkt__str_strkt_id aus Spalte prj.pl_psgefstrkt.str_strkt_id

fk__prj_pp_protokoll__prj_personen_id aus Spalte prj.pp_prtprotokoll.prj_personen_id

fk__prj_pp_protokoll__prj_projekte_id aus Spalte prj.pp_prtprotokoll.prj_projekte_id

fk__prj_pp_protokoll__prt_protokoll_id aus Spalte prj.pp_prtprotokoll.prt_protokoll_id

fk__prj_pp_protokoll__spr_uebvar_id aus Spalte prj.pp_prtprotokoll.spr_uebvar_id

fk__prj_pp_psgefszenario__prj_personen_id aus Spalte prj.pp_psgefszenario.prj_personen_id

fk__prj_pp_psgefszenario__prj_projekte_id aus Spalte prj.pp_psgefszenario.prj_projekte_id

fk__prj_pp_psgefszenario__ps_gefszenario_id aus Spalte prj.pp_psgefszenario.ps_gefszenario_

fk__prj_pp_psgefszenario__spr_uebvar_id aus Spalte prj.pp_psgefszenario.spr_uebvar_id

fk__prj_pp_psgefszenario__spr_uebvar_id_reprgefszen aus Spalte prj.pp_psgefszenario.spr_ueb

fk__prj_pp_psgefszenario__spr_uebvar_id_ursp aus Spalte prj.pp_psgefszenario.spr_uebvar_id_u

fk__prj_pp_strstrkt__prj_personen_id aus Spalte prj.pp_strstrkt.prj_personen_id

fk__prj_pp_strstrkt__prj_projekte_id aus Spalte prj.pp_strstrkt.prj_projekte_id

fk__prj_pp_strstrkt__spr_uebvar_id aus Spalte prj.pp_strstrkt.spr_uebvar_id

fk__prj_pp_strstrkt__spr_uebvar_id_zus aus Spalte prj.pp_strstrkt.spr_uebvar_id_zus

fk__prj_pp_strstrkt__str_strkt_id aus Spalte prj.pp_strstrkt.str_strkt_id

fk__prj_ppl_dokdoku__dok_doku_id aus Spalte prj.ppl_dokdoku.dok_doku_id

fk__prj_ppl_dokdoku__prj_personen_id aus Spalte prj.ppl_dokdoku.prj_personen_id

fk__prj_ppl_dokdoku__prj_projekte_id aus Spalte prj.ppl_dokdoku.prj_projekte_id

Unique Constraints

uc__prj_p_dokstrkt in Tabelle prj.p_dokstrkt für die Spalte(n) prj.p_dokstrkt.idfrei, prj.p_dokstrkt.dok_doku_id, prj.p_dokstrkt.prj_projekte_id, prj.p_dokstrkt.str_strk

uc__prj_p_prtstrstrkt in Tabelle prj.p_prtstrstrkt für die Spalte(n) prj.p_prtstrstrkt.idfrei, prj.p_prtstrstrkt.prj_projekte_id, prj.p_prtstrstrkt.prt_protokoll_id, prj.p_prtstr

uc__prj_p_prtverbindung in Tabelle prj.p_prtverbindung für die Spalte(n) prj.p_prtverbindung.id prj.p_prtverbindung.prj_projekte_id, prj.p_prtverbindung.prt_protokoll_id_nachf, prj.p_prtverbindung.prt_protokoll_id_vorg

uc__prj_p_psabk in Tabelle prj.p_psabk für die Spalte(n) prj.p_psabk.idfrei, prj.p_psabk.prj_pr prj.p_psabk.ps_abk_id

uc__prj_p_strverbindung in Tabelle prj.p_strverbindung für die Spalte(n) prj.p_strverbindung.idfrei
prj.p_strverbindung.prj_projekte_id, prj.p_strverbindung.str_strkt_id_nachf,
prj.p_strverbindung.str_strkt_id_vorg

uc__prj_personen in Tabelle prj.personen für die Spalte(n) prj.personen.idfrei, prj.personen.name_
prj.personen.name_v, prj.personen.intern_firma_id

uc__prj_pl_dokausgabe in Tabelle prj.pl_dokausgabe für die Spalte(n) prj.pl_dokausgabe.idfrei,
prj.pl_dokausgabe.logentfernt, prj.pl_dokausgabe.logdatuhrz, prj.pl_dokausgabe.dok_dok
prj.pl_dokausgabe.prj_projekte_id

uc__prj_pl_prtdokdoku in Tabelle prj.pl_prtdokdoku für die Spalte(n) prj.pl_prtdokdoku.idfrei,
prj.pl_prtdokdoku.logentfernt, prj.pl_prtdokdoku.logdatuhrz, prj.pl_prtdokdoku.dok_dok
prj.pl_prtdokdoku.prj_projekte_id, prj.pl_prtdokdoku.prt_protokoll_id, prj.pl_prtdokdok

uc__prj_pl_prtpsgefszenario in Tabelle prj.pl_prtpsgefszenario für die Spalte(n) prj.pl_prtpsgefsze
prj.pl_prtpsgefszenario.prj_projekte_id, prj.pl_prtpsgefszenario.prt_protokoll_id,
prj.pl_prtpsgefszenario.ps_gefszenario_id, prj.pl_prtpsgefszenario.logentfernt,
prj.pl_prtpsgefszenario.logdatuhrz

uc__prj_pl_psgefabk in Tabelle prj.pl_psgefabk für die Spalte(n) prj.pl_psgefabk.idfrei,
prj.pl_psgefabk.prj_projekte_id, prj.pl_psgefabk.ps_abk_id, prj.pl_psgefabk.ps_gefszena
prj.pl_psgefabk.logentfernt, prj.pl_psgefabk.logdatuhrz

uc__prj_pl_psgefanm in Tabelle prj.pl_psgefanm für die Spalte(n) prj.pl_psgefanm.idfrei,
prj.pl_psgefanm.prj_projekte_id, prj.pl_psgefanm.ps_anmerkung_id, prj.pl_psgefanm.ps_ge
prj.pl_psgefanm.logentfernt, prj.pl_psgefanm.logdatuhrz

uc__prj_pl_psgefstrkt in Tabelle prj.pl_psgefstrkt für die Spalte(n) prj.pl_psgefstrkt.idfrei,
prj.pl_psgefstrkt.prj_projekte_id, prj.pl_psgefstrkt.ps_gefszenario_id, prj.pl_psgefstr
prj.pl_psgefstrkt.logentfernt, prj.pl_psgefstrkt.logdatuhrz

uc__prj_pp_protokoll in Tabelle prj.pp_prtprotokoll für die Spalte(n) prj.pp_prtprotokoll.idfrei,
prj.pp_prtprotokoll.prj_projekte_id, prj.pp_prtprotokoll.prt_protokoll_id, prj.pp_prtpr
prj.pp_prtprotokoll.prj_personen_id

uc__prj_pp_psgefszenario in Tabelle prj.pp_psgefszenario für die Spalte(n) prj.pp_psgefszenario.idf
prj.pp_psgefszenario.prj_projekte_id, prj.pp_psgefszenario.ps_gefszenario_id

uc__prj_pp_strstrkt in Tabelle prj.pp_strstrkt für die Spalte(n) prj.pp_strstrkt.idfrei,
prj.pp_strstrkt.prj_personen_id, prj.pp_strstrkt.prj_projekte_id, prj.pp_strstrkt.str_s

uc__prj_ppl_dokdoku in Tabelle prj.ppl_dokdoku für die Spalte(n) prj.ppl_dokdoku.idfrei,
prj.ppl_dokdoku.logentfernt, prj.ppl_dokdoku.logdatuhrz, prj.ppl_dokdoku.dok_doku_id,
prj.ppl_dokdoku.prj_personen_id, prj.ppl_dokdoku.prj_projekte_id

uc__prj_projekte in Tabelle prj.projekte für die Spalte(n) prj.projekte.idfrei, prj.projekte.prjna

Check Constraints

cc__prj_p_strverbindung in Tabelle prj.p_strverbindung die Bedingung (((str_strkt_id_nachf = 0) AND (str_strkt_id_vorg = 0)) OR ((id > 0) AND (str_strkt_id_vorg > 0)))

cc__prj_pp_protokoll in Tabelle prj.pp_prtprotokoll die Bedingung ((((((((((id = 0) AND (idfrei = 0)) AND (prj_personen_id = 0)) AND (prj_projekte_id = 0)) AND (prt_protokoll_id = 0)) AND (spr_uebvar_id = 0)) OR ((id > 0) AND (idfrei > 0))) OR (((((id > 0) AND (idfrei = 0)) AND (prj_personen_id = 0)) AND (prj_projekt > 0)) AND (prt_protokoll_id > 0))) OR (((((((id > 0) AND (idfrei = 0)) AND (prj_personen_id > 0)) AND (prj_projekte_id > 0)) AND (prt_protokoll_id > 0)) AND (spr_uebvar_id = 0)) AND (teilnehmer = true))) OR (((((((id > 0) AND (idfrei = 0)) AND (prj_personen_id > 0)) AND (prj_projekte_id > 0)) AND (prt_protokoll_id > 0)) AND (spr_uebvar_id = 0)) AND (teilnehmer = false)) AND (termin IS NOT NULL)))

5.10 prj.nimm__tabwerte

Funktionsname

prj.nimm__tabwerte(IN tabname varchar, IN tabwerte varchar[], OUT tabid varchar)

Funktionsparameter/Rückgabewert(e)

tabname Name der Tabelle (ohne Schemanamen) des Schemas prj, welche die Werte übernehmen soll (Details siehe Funktionsbeschreibung — Werteliste einlesen).

tabwerte Liste der Werte, welche in die aktuelle Tabelle eingelesen werden sollen (Details siehe Funktionsbeschreibung — Werteliste einlesen).

tabid Es wird die Identifikationsnummer des aktuell eingelesenen Wertes prj.*.id zurückgegeben.

Funktionsbeschreibung

Mittels dieser Funktion werden in die Tabellen des Schemas prj die Werte eingelesen.

Werteliste einlesen:
Dem Funktionsparameter tabwerte ist eine Werteliste (ARRAY) zu übergeben (die maximale Anzahl und die Reihenfolge der Werte ist fest vorgegeben — siehe oben). Soll nur eine Teilmenge der Werte übergeben werden, dann sind alle Werte von links nach rechts zu spezifizieren (nichtbenötigte Werte am Ende der Liste dürfen weggelassen werden).

personen

1. intern_firma_id
2. name_n
3. name_v
4. notizen

Es müssen mindestens die ersten drei Werte übergeben werden. Der erste Wert muß größer `0` sein.

projekte

1. `prjname`
2. `prjbeschr`
3. `makefile`
4. `notizen`

Es müssen mindestens die ersten beiden Werte übergeben werden.

p_dokstrkt, p_prtstrstrkt, p_prtverbindung,
p_psabk, p_strverbindung

1. `prj_projekte_id`
2. `dok_doku_id, prt_protokoll_id, prt_protokoll_id_nachf,`
   `ps_abk_id, str_strkt_id_vorg`
3. `dok_strkt_id, str_strkt_id, prt_protokoll_id_vorg,`
   `spr_uebvar_id, str_strkt_id_nachf`
4. `notizen, notizen, notizen,`
   `spr_uebvar_id_l, fbtor`
5. `---, ---, ---,`
   `spr_uebvar_id_zus, notizen`
6. `---, ---, ---,`
   `notizen, ---`

Es müssen mindestens die ersten drei Werte übergeben werden. `prj_projekte_id` darf `0` sein, die anderen beiden Werte müssen größer `0` sein.

pl_dokausgabe, pl_prtdokdoku, pl_prtpsgefszenario,
pl_psgefabk, pl_psgefanm, pl_psgefstrkt

1. `prj_projekte_id`
2. `dok_doku_id, dok_doku_id, ps_gefszenario_id,`
   `ps_gefszenario_id, ps_gefszenario_id, ps_gefszenario_id`
3. `logentfernt, prt_protokoll_id, prt_protokoll_id,`
   `logentfernt, logentfernt, str_strkt_id`
4. `stand, nr, notizen,`
   `ps_abk_id, ps_anmerkung_id, notizen`

5. `datum, spr_uebvar_id, logentfernt,`
   `spr_uebvar_id_zus, spr_uebvar_id, logentfernt`

6. `istfrei, logentfernt, logdatuhrz,`
   `notizen, notizen, logdatuhrz`

7. `notizen, spr_uebvar_id_zus, logdquelle,`
   `logdatuhrz, logdatuhrz, logdquelle`

8. `logdatuhrz, notizen, logname,`
   `logdquelle, logdquelle, logname`

9. `logdquelle, logdatuhrz, logdqok,`
   `logname, logname, logdqok`

10. `logname, logdquelle, ---,`
    `logdqok, logdqok, ---`

11. `logdqok, logname, ---,`
    `---, ---, ---`

12. `---, logdqok, ---,`
    `---, ---, ---`

Es müssen mindestens die ersten beiden Werte übergeben werden. `prj_projekte_id` darf `0` sein, der andere Wert muß größer `0` sein. Bei diesen Tabellen (Tabellen mit Logbuchfunktion) wird nicht nach bereits eingelesenen Einträgen gesucht. Somit ist es möglich mehrfach dieselben Einträge einzulesen.

pp_prtprotokoll, pp_prtprotokoll_pers, pp_psgefszenario, pp_strstrkt, ppl_dokdoku

1. `prj_projekte_id`

2. `prt_protokoll_id, prt_protokoll_id, ps_gefszenario_id,`
   `str_strkt_id, dok_doku_id`

3. `spr_uebvar_id, teilnehmer, prj_personen_id,`
   `prj_personen_id, prj_personen_id`

4. `datuhrz, prj_personen_id, spr_uebvar_id,`
   `spr_uebvar_id, logentfernt`

5. `intern, termin, spr_uebvar_id_reprgefszen,`
   `spr_uebvar_id_zus, notizen` Für die Spalten `spr_uebvar_id_fzg` und `spr_uebvar_id` muß der Wert größer `0` sein.

6. `erledigt, duanfrage, spr_uebvar_id_ursp,`
   `fbereignis, logdatuhrz`

7. notizen, notizen, notizen,
   notizen, logdquelle

8. ---, ---, ---,
   ---, logname

9. ---, ---, ---,
   ---, logdqok

Bei den Tabellen ppl_dokdoku müssen die ersten drei und bei den Tabellen pp_psgefszenario, pp_strstrkt die ersten zwei Werte übergeben werden. prj_projekte_id darf 0 sein, die anderen Drei bzw. Zwei Werte müssen größer 0 sein.
Bei der Tabelle pl_prtdokdoku müssen die ersten fünf Werte übergeben werden.

Die Tabelle pp_prtprotokoll deckt drei Aspekte ab:

1. Protokolleintrag einem Projekt zuordnen, projektspezifisch modifizieren und Randbedingungen setzen (pp_prtprotokoll wählen);

2. dem Protokolleintrag die beteiligten Personen zuordnen (pp_prtprotokoll_pers wählen);

3. dem Protokolleintrag der eine Frage / Aufgabe ist, die relevanten Informationen zuordnen (pp_prtprotokoll_pers wählen);

wobei ein Protokolleintrag ein Gefahrenprotokolleintrag, eine Sicherheitsüberprüfung oder ein Terminplaneintrag (je nach Sinnfälligkeit) sein kann.
Zur Unterscheidung der drei Aspekte werden die Spalten prj_personen_id und teilnehmer verwendet. Es gilt für

1. prj_personen_id = 0 und teilnehmer = NULL

2. prj_personen_id > 0 und teilnehmer = true

3. prj_personen_id > 0 und teilnehmer = false

Die Spalten der Tabelle pp_prtprotokoll werden zur Abbildung der drei Aspekte wie folgt verwendet:

```
notizen ........... 1
datuhrz ........... 1
duanfrage ............. 3
erledigt .......... 1
intern ............ 1
prj_personen_id ..... 2 3
prj_projekte_id ... 1 2 3
prt_protokoll_id .. 1 2 3
spr_uebvar_id ..... 1
teilnehmer .......... 2 3
termin ................ 3
```

Beispiele

```
ACTUNG ! Die Beispiele sind nicht aktualisiert --- muß später bereinit werden.

-- Fehlermeldung. Ungültige Tabelle !
SELECT * FROM prj.nimm__tabwerte( NULL, NULL );

-- Fehlermeldung. Bei allen Tabellen müssen mindestens zwei Elemente übergeben
werden
SELECT * FROM prj.nimm__tabwerte( 'projekte', ARRAY['1'] );

-- Fehlermeldung. Die übergebenen Parameter müssen größer 0 sein.
SELECT * FROM prj.nimm__tabwerte( 'p_dokkomp', ARRAY['0', '1'] );
SELECT * FROM prj.nimm__tabwerte( 'p_dokkomp', ARRAY['1', '0'] );
SELECT * FROM prj.nimm__tabwerte( 'pl_psgefanm', ARRAY['1', '1', '0'] );
SELECT * FROM prj.nimm__tabwerte( 'ppu_kmpkomp', ARRAY['1', '1', '1', '', '0'] );
SELECT * FROM prj.nimm__tabwerte( 'pp_prtprotokoll', ARRAY['1', '1', '1', '', '',
'', '', '', '', '', '0'] );

-- Fehlermeldung. Es dürfen nur die zulässigen Begriffe verwendet werden
SELECT * FROM prj.nimm__tabwerte( 'pp_strstrkt', ARRAY['1', '22', '1', '99', 'ttt']
);
SELECT * FROM prj.nimm__tabwerte( 'p_strverbindung', ARRAY['1', '1', 'ttt'] );

-- prj.projekte einlesen
SELECT * FROM prj.projekte;
SELECT * FROM prj.nimm__tabwerte( 'projekte', ARRAY['testProj', 'Testprojekt',
'Inhalt des makefile', 'Zusatzinformationen'] );
SELECT * FROM prj.projekte;

-- dok.* einlesen
SELECT * FROM prj.pp_dokausgabe;
SELECT * FROM prj.nimm__tabwerte( 'pp_dokausgabe', ARRAY[prj.nimm__tabwerte( 'projekte'
ARRAY['testProj', 'Testprojekt'] ), dok.nimm__tabwerte( 'ausgabe', ARRAY[ dok.nimm__dok
ARRAY['08-15', '1'] ), '01'] ), prj.nimm__tabwerte( 'personen', ARRAY['1', 'Bellair',
'Michael'] ), 'Zusatzinformationen'] );
SELECT * FROM prj.pp_dokausgabe;
SELECT * FROM prj.p_dokkomp;
SELECT * FROM prj.nimm__tabwerte( 'p_dokkomp', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), dok.nimm__tabwerte( 'komp', ARRAY[dok.nimm__doku(
ARRAY['08-15', '1'] ), kmp.nimm__komp( ARRAY['1', '47/11'] )] ), 'Zusatzinformationen'
);
SELECT * FROM prj.p_dokkomp;
SELECT * FROM prj.p_dokstrkt;
SELECT * FROM prj.nimm__tabwerte( 'p_dokstrkt', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), dok.nimm__tabwerte( 'strkt', ARRAY[dok.nimm__doku(
ARRAY['08-15', '1'] ), str.nimm__strkt( ARRAY['1', '-BAAA', spr.nimm__wgruppe( 'Struktu
```

```
'1', '0' )] )] ), 'Zusatzinformationen'] );
SELECT * FROM prj.p_dokstrkt;

-- kmp.* einlesen
SELECT * FROM prj.ppu_kmpkomp;
SELECT * FROM prj.nimm__tabwerte( 'ppu_kmpkomp', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), kmp.nimm__komp( ARRAY['1', '08-15'] ), prj.nimm__tabwer
'personen', ARRAY['1', 'Bellair', 'Michael'] ), 'kmp-id-fzg', spr.nimm__wgruppe(
'Komponentenbezeichnung-Fahrezeughersteller', '1', '0' ), 'Zusatzinformationen']
);
SELECT * FROM prj.ppu_kmpkomp;
SELECT * FROM prj.p_kmpstrkt;
SELECT * FROM prj.nimm__tabwerte( 'p_kmpstrkt', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), kmp.nimm__strkt( ARRAY[kmp.nimm__komp( ARRAY['1',
'08-15'] ), str.nimm__strkt( ARRAY['1', '-BAAA', spr.nimm__wgruppe( 'Strukturpunkt',
'1', '0' )] ), 'Zusatzinformationen'] ), 'Zusatzinformationen'] );
SELECT * FROM prj.p_kmpstrkt;

-- prt.* einlesen
SELECT * FROM prj.p_prtdokausgabe;
SELECT * FROM prj.nimm__tabwerte( 'p_prtdokausgabe', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), prt.nimm__tabwerte( 'dokausgabe', ARRAY[prt.nimm__proto
ARRAY[spr.nimm__wgruppe( 'Aktueller Protokolleintrag.', '1', '0' ), 'PE_OK'] ),
dok.nimm__tabwerte( 'ausgabe', ARRAY[ dok.nimm__doku( ARRAY['08-15', '1'] ), '01']
), 'Zusatzinformationen'] ), 'Zusatzinformationen'] );
SELECT * FROM prj.p_prtdokausgabe;
SELECT * FROM prj.p_prtkomp;
SELECT * FROM prj.nimm__tabwerte( 'p_prtkomp', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), prt.nimm__tabwerte( 'komp', ARRAY[prt.nimm__protokollei
ARRAY[spr.nimm__wgruppe( 'Aktueller Protokolleintrag.', '1', '0' ), 'PE_OK'] ),
kmp.nimm__komp( ARRAY['1', '47/11'] )] ), 'Zusatzinformationen'] );
SELECT * FROM prj.p_prtkomp;
SELECT * FROM prj.pp_prtprotokoll;
SELECT * FROM prj.nimm__tabwerte( 'pp_prtprotokoll', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), prt.nimm__protokolleintrag( ARRAY[spr.nimm__wgruppe(
'Aktueller Protokolleintrag.', '1', '0' ), 'PE_OK'] ), prj.nimm__tabwerte( 'personen',
ARRAY['1', 'Bellair', 'Michael'] ), 'true', '2007-06-08', '2007-06-20', '2007-07-07',
'true', 'true', spr.nimm__wgruppe( 'Projektabhängiger, aktueller Protokolleintrag.',
'1', '0' ), 'Zusatzinformationen'] );
SELECT * FROM prj.pp_prtprotokoll;
SELECT * FROM prj.p_prtstrstrkt;
SELECT * FROM prj.nimm__tabwerte( 'p_prtstrstrkt', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), prt.nimm__tabwerte( 'strkt', ARRAY[prt.nimm__protokolle
ARRAY[spr.nimm__wgruppe( 'Aktueller Protokolleintrag.', '1', '0' ), 'PE_OK'] ),
str.nimm__strkt( ARRAY['1', '-BAAA', spr.nimm__wgruppe( 'Strukturpunkt', '1', '0'
)])]), 'Zusatzinformationen'] );
```

```
SELECT * FROM prj.p_prtstrstrkt;
SELECT * FROM prj.p_prtverbindung;
SELECT * FROM prj.nimm__tabwerte( 'p_prtverbindung', ARRAY[prj.nimm__tabwerte( 'projekt
ARRAY['testProj', 'Testprojekt'] ), prt.nimm__tabwerte( 'verbindung', ARRAY[prt.nimm__p
ARRAY[spr.nimm__wgruppe( 'Aktueller Protokolleintrag.', '1', '0' ), 'PE_OK'] ),
prt.nimm__protokolleintrag( ARRAY[spr.nimm__wgruppe( 'Zugeordneter Aktueller Protokolle
'1', '0' ), 'PE_OK'] )]), 'Zusatzinformationen'] );
SELECT * FROM prj.p_prtverbindung;

-- ps.* einlesen
SELECT * FROM prj.p_psabk;
SELECT * FROM prj.nimm__tabwerte( 'p_psabk', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), ps.nimm__abk( ARRAY[spr.nimm__wgruppe( 'abk',
'1', '0' ), spr.nimm__wgruppe( 'Abkürzung', '1', '0' ), 'BP'] ), spr.nimm__wgruppe(
'projabk', '1', '0' ), spr.nimm__wgruppe( 'projektabhängige Abkürzung', '1', '0'
), 'Zusatzinformationen'] );
SELECT * FROM prj.p_psabk;
SELECT * FROM prj.pl_psgefabk;
SELECT * FROM prj.nimm__tabwerte( 'pl_psgefabk', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), ps.nimm__aspekte( 'gefabk', ARRAY[ ps.nimm__gefszen
ARRAY[spr.nimm__wgruppe( 'Gefahrenszenario', '1', '0' ), spr.nimm__wgruppe( 'Ursprung
des Gefahrenszenarios', '1', '0' )] ), ps.nimm__abk( ARRAY[spr.nimm__wgruppe( 'abk',
'1', '0' ), spr.nimm__wgruppe( 'Abkürzung', '1', '0' ), 'BP'] )] ), '2007-06-06',
'datenquelle', 'micha', 'true', 'Zusatzinformationen'] );
SELECT * FROM prj.pl_psgefabk;
SELECT * FROM prj.pl_psgefanm;
SELECT * FROM prj.nimm__tabwerte( 'pl_psgefanm', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), ps.nimm__aspekte( 'gefanm', ARRAY[ps.nimm__gefszena
ARRAY[spr.nimm__wgruppe( 'Gefahrenszenario', '1', '0' ), spr.nimm__wgruppe( 'Ursprung
des Gefahrenszenarios', '1', '0' )] ), spr.nimm__wgruppe( 'Anmerkungen', '1', '0'
)] ), spr.nimm__wgruppe( 'projektspezifische Anmerkungen', '1', '0' ), 'true',
'2007-06-06', 'datenquelle', 'micha', 'true', 'Zusatzinformationen'] );
SELECT * FROM prj.pl_psgefanm;
SELECT * FROM prj.pl_psgefkomp;
SELECT * FROM prj.nimm__tabwerte( 'pl_psgefkomp', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), ps.nimm__aspekte( 'gefkomp', ARRAY[ps.nimm__gefszen
ARRAY[spr.nimm__wgruppe( 'Gefahrenszenario', '1', '0' ), spr.nimm__wgruppe( 'Ursprung
des Gefahrenszenarios', '1', '0' )] ), kmp.nimm__komp( ARRAY['1','08-15'] )] ),
'2007-06-06', 'datenquelle', 'micha', 'true', 'Zusatzinformationen'] );
SELECT * FROM prj.pl_psgefkomp;
SELECT * FROM prj.pl_psgefnetz;
SELECT * FROM prj.nimm__tabwerte( 'pl_psgefnetz', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), ps.nimm__aspekte( 'gefnetz', ARRAY[ps.nimm__gefszen
ARRAY[spr.nimm__wgruppe( 'Gefahrenszenario', '1', '0' ), spr.nimm__wgruppe( 'Ursprung
des Gefahrenszenarios', '1', '0' )]), str.nimm__strkt( ARRAY['1', '-BAAA', spr.nimm__wg
'Strukturpunkt', '1', '0' )])] ), '2007-06-06', 'datenquelle', 'micha', 'true',
```

```
'Zusatzinformationen'] );
SELECT * FROM prj.pl_psgefnetz;
SELECT * FROM prj.pl_prtpsgefszenario;
SELECT * FROM prj.nimm__tabwerte( 'pl_prtpsgefszenario', ARRAY[prj.nimm__tabwerte(
'projekte', ARRAY['testProj', 'Testprojekt'] ), ps.nimm__aspekte( 'gefprot', ARRAY[ps.nimm
ARRAY[spr.nimm__wgruppe( 'Gefahrenszenario', '1', '0' ), spr.nimm__wgruppe( 'Ursprung
des Gefahrenszenarios', '1', '0' )]), prt.nimm__protokolleintrag( ARRAY[spr.nimm__wgruppe(
'Zugeordneter Aktueller Protokolleintrag.', '1', '0' ), 'PE_OK'] )] ), 'true',
'2007-06-06', 'datenquelle', 'micha', 'true', 'Zusatzinformationen'] );
SELECT * FROM prj.pl_prtpsgefszenario;
SELECT * FROM prj.pl_psgefstrkt;
SELECT * FROM prj.nimm__tabwerte( 'pl_psgefstrkt', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), ps.nimm__aspekte( 'gefstrkt', ARRAY[ps.nimm__gefszenari
ARRAY[spr.nimm__wgruppe( 'Gefahrenszenario', '1', '0' ), spr.nimm__wgruppe( 'Ursprung
des Gefahrenszenarios', '1', '0' )]), str.nimm__strkt( ARRAY['1', '-BAAA', spr.nimm__wgrupp
'Strukturpunkt', '1', '0' )])] ), 'true', '2007-06-06', 'datenquelle', 'micha',
'true', 'Zusatzinformationen'] );
SELECT * FROM prj.pl_psgefstrkt;
SELECT * FROM prj.pp_psgefszenario;
SELECT * FROM prj.nimm__tabwerte( 'pp_psgefszenario', ARRAY[prj.nimm__tabwerte(
'projekte', ARRAY['testProj', 'Testprojekt'] ), ps.nimm__gefszenario( ARRAY[spr.nimm__wgrup
'Gefahrenszenario', '1', '0' ), spr.nimm__wgruppe( 'Ursprung des Gefahrenszenarios',
'1', '0' )]), prj.nimm__tabwerte( 'personen', ARRAY['1', 'Bellair', 'Michael']
), spr.nimm__wgruppe( 'projektspezifisch formuliertes Gefahrenszenario', '1', '0'
), spr.nimm__wgruppe( 'projektspezifisch formulierter Ursprung des Gefahrenszenarios',
'1', '0' ), 'Zusatzinformationen'] );
SELECT * FROM prj.pp_psgefszenario;

-- str.* einlesen
SELECT * FROM prj.pp_strstrkt;
SELECT * FROM prj.nimm__tabwerte( 'pp_strstrkt', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), str.nimm__strkt( ARRAY['1', '-BAAA', spr.nimm__wgruppe(
'Strukturpunkt', '1', '0' )]), prj.nimm__tabwerte( 'personen', ARRAY['1', 'Bellair',
'Michael'] ), spr.nimm__wgruppe( 'Strukturpunkt übersetzt ?', '1', '0' ), 'HAUS',
'Zusatzinformationen'] );
SELECT * FROM prj.pp_strstrkt;
SELECT * FROM prj.p_strverbindung;
SELECT * FROM prj.nimm__tabwerte( 'p_strverbindung', ARRAY[prj.nimm__tabwerte( 'projekte',
ARRAY['testProj', 'Testprojekt'] ), str.nimm__verbindung( ARRAY[str.nimm__strkt(
ARRAY['1', '-BAAA1', spr.nimm__wgruppe( 'Strukturpunkt-1', '1', '0' )]), str.nimm__strkt(
ARRAY['1', '-BAAA2', spr.nimm__wgruppe( 'Strukturpunkt-2', '1', '0' )])] ), 'UND',
'Zusatzinformationen'] );
SELECT * FROM prj.p_strverbindung;
```

5.11 prj.zeige__dokdoku

Beschreibung der Spalten

dok__doku__id siehe dok.doku.id

dok__doku__aspekt siehe dok.doku.aspekt

dok__doku__notizen siehe dok.doku.notizen

dok__doku__intern_firma_id siehe dok.doku.intern_firma_id

dok__doku__spr_uebvar_id_kb siehe dok.doku.spr_uebvar_id_kb

prj__pl_prtdokdoku__id siehe prj.pl_prtdokdoku.id

prj__pl_prtdokdoku__prj_projekte_id siehe prj.pl_prtdokdoku.prj_projekte_id

prj__pl_prtdokdoku__prt_protokoll_id siehe prj.pl_prtdokdoku.prt_protokoll_id

prj__pl_prtdokdoku__nr siehe prj.pl_prtdokdoku.nr

prj__pl_prtdokdoku__spr_uebvar_id siehe prj.pl_prtdokdoku.spr_uebvar_id

prj__pl_prtdokdoku__spr_uebvar_id_zus siehe prj.pl_prtdokdoku.spr_uebvar_id_zus

prj__pl_prtdokdoku__notizen siehe prj.pl_prtdokdoku.notizen

prj__pl_prtdokdoku__logdatuhrz siehe prj.pl_prtdokdoku.logdatuhrz

prj__pl_prtdokdoku__logdqok siehe prj.pl_prtdokdoku.logdqok

prj__pl_prtdokdoku__logdquelle siehe prj.pl_prtdokdoku.logdquelle

prj__pl_prtdokdoku__logentfernt siehe prj.pl_prtdokdoku.logentfernt

prj__pl_prtdokdoku__logname siehe prj.pl_prtdokdoku.logname

prj__pl_dokausgabe__id siehe prj.pl_dokausgabe.id

prj__pl_dokausgabe__prj_projekte_id siehe prj.pl_dokausgabe.prj_projekte_id

prj__pl_dokausgabe__stand siehe prj.pl_dokausgabe.stand

prj__pl_dokausgabe__datum siehe prj.pl_dokausgabe.datum

prj__pl_dokausgabe__istfrei siehe prj.pl_dokausgabe.istfrei

prj__pl_dokausgabe__notizen siehe prj.pl_dokausgabe.notizen

prj__pl_dokausgabe__logdatuhrz siehe prj.pl_dokausgabe.logdatuhrz

prj__pl_dokausgabe__logdqok siehe prj.pl_dokausgabe.logdqok

prj__pl_dokausgabe__logdquelle siehe prj.pl_dokausgabe.logdquelle

prj__pl_dokausgabe__logentfernt siehe prj.pl_dokausgabe.logentfernt

prj__pl_dokausgabe__logname siehe prj.pl_dokausgabe.logname

prj__ppl_dokdoku__id siehe prj.ppl_dokdoku.id

prj__ppl_dokdoku__prj_personen_id siehe prj.ppl_dokdoku.prj_personen_id

prj__ppl_dokdoku__prj_projekte_id siehe prj.ppl_dokdoku.prj_projekte_id

prj__ppl_dokdoku__notizen siehe prj.ppl_dokdoku.notizen

prj__ppl_dokdoku__logdatuhrz siehe prj.ppl_dokdoku.logdatuhrz

prj__ppl_dokdoku__logdqok siehe prj.ppl_dokdoku.logdqok

prj__ppl_dokdoku__logdquelle siehe prj.ppl_dokdoku.logdquelle

prj__ppl_dokdoku__logentfernt siehe prj.ppl_dokdoku.logentfernt

prj__ppl_dokdoku__logname siehe prj.ppl_dokdoku.logname

prj__p_dokstrkt__id siehe prj.p_dokstrkt.id

prj__p_dokstrkt__notizen siehe prj.p_dokstrkt.notizen

prj__p_dokstrkt__prj_projekte_id siehe prj_projekte_id

prj__p_dokstrkt__str_strkt_id siehe prj.p_dokstrkt.str_strkt_id

Beschreibung der Sicht
Es werden die Dokumente (siehe dok.doku, prj.pl_prtdokdoku, prj.pl_dokausgabe, prj.ppl_dokdoku) angezeigt.

5.12 prj.zeige__p_prtverbindung

5.13 prj.zeige__pl_dokausgabe_le

Beschreibung der Spalten

prj__pl_dokausgabe__id siehe prj.pl_dokausgabe.id

Beschreibung der Sicht
Zu jedem in der Tabelle prj.pl_dokausgabe enthaltenen Dokument wird unter Berücksichtigung des Projektes die Identifikationsnummer prj.pl_dokausgabe.id der jeweils letzten zugeordneten Ausgabe angezeigt (Logbuchfunktion).

5.14 prj.zeige__pl_prtdokdoku_le

Beschreibung der Spalten

prj__pl_prtdokdoku__id siehe prj.pl_prtdokdoku.id

Beschreibung der Sicht

Zu jeder Sicherheitsanforderung, Protokolleintrag, Terminplaneintrag (siehe Tabelle prt.protokoll) werden unter Berücksichtigung des Projektes prj.pl_prtdokdoku.id und der Zusatzinformationen prj.pl_prtdokdoku.spr_uebvar_id_zus die zugeordneten Dokumente (siehe dok.doku) angezeigt (Logbuchfunktion).

5.15 prj.zeige__pl_prtpsgefszenario

Beschreibung der Spalten

id siehe prj.pl_prtpsgefszenario.id

notizen siehe prj.pl_prtpsgefszenario.notizen

logdatuhrz siehe prj.pl_prtpsgefszenario.logdatuhrz

logdqok siehe prj.pl_prtpsgefszenario.logdqok

logdquelle siehe prj.pl_prtpsgefszenario.logdquelle

logentfernt siehe prj.pl_prtpsgefszenario.logentfernt

logname siehe prj.pl_prtpsgefszenario.logname

prj_projekte_id siehe prj.pl_prtpsgefszenario.prj_projekte_id

prt_protokoll_id siehe prj.pl_prtpsgefszenario.prt_protokoll_id

ps_gefszenario_id siehe prj.pl_prtpsgefszenario.ps_gefszenario_id

Beschreibung der Sicht

Es werden die projektspezifischen Sicherheitsanforderungen, Gefahrenprotokolleinträge bzw. Terminplaneinträge angezeigt (siehe prj.pl_prtpsgefszenario).

5.16 prj.zeige__pl_prtpsgefszenario_le

Beschreibung der Spalten

prj__pl_prtpsgefszenario__id siehe prj.pl_prtpsgefszenario.id

Beschreibung der Sicht

Zu jedem in der Tabelle prj.pl_prtpsgefszenario enthaltenen Aspekt wird der jeweils letzte zugeordnete in ps verwendete Protokolleintrag angezeigt (Logbuchfunktion). Die unterschiedlichen Aspekte, zu denen das Logbuch geführt wird, werden anhand der Spalten prj.pl_prtpsgefszenario.prj_projekte_id, prj.pl_prtpsgefszenario.prt_protokoll_id, prj.pl_prtpsgefszenario.ps_gefszenario_id unterschieden.

5.17 prj.zeige__pl_psgefabk

Beschreibung der Spalten

id siehe prj.pl_psgefabk.id

notizen siehe prj.pl_psgefabk.notizen

logdatuhrz siehe prj.pl_psgefabk.logdatuhrz

logdqok siehe prj.pl_psgefabk.logdqok

logdquelle siehe prj.pl_psgefabk.logdquelle

logentfernt siehe prj.pl_psgefabk.logentfernt

logname siehe prj.pl_psgefabk.logname

prj_projekte_id siehe prj.pl_psgefabk.prj_projekte_id

ps_abk_id siehe prj.pl_psgefabk.ps_abk_id

ps_gefszenario_id siehe prj.pl_psgefabk.ps_gefszenario_id

spr_uebvar_id_zus siehe prj.pl_psgefabk.spr_uebvar_id_zus

Beschreibung der Sicht

Es werden die projektspezifischen (siehe prj.pl_psgefabk) Abkürzungen angezeigt, die den Gefahrenszenarien zugeordnet sind.

5.18 prj.zeige__pl_psgefabk_le

Beschreibung der Spalten

prj__prj.pl_psgefabk__id siehe prj.pl_psgefabk.id

Beschreibung der Sicht

Zu jedem in der Tabelle prj.pl_psgefabk enthaltenen Aspekt wird die jeweils letzte zugeordnete in ps verwendete Abkürzung angezeigt (Logbuchfunktion). Die unterschiedlichen Aspekte, zu denen das Logbuch geführt wird, werden anhand der Spalten prj.pl_psgefabk.prj_projekte_id, prj.pl_psgefabk.ps_gefszenario_id, prj.pl_psgefabk.ps_abk_id unterschieden.

5.19 prj.zeige__pl_psgefanm

Beschreibung der Spalten

id siehe prj.pl_psgefanm.id

notizen siehe prj.pl_psgefanm.notizen

logdatuhrz siehe prj.pl_psgefanm.logdatuhrz

logdqok siehe prj.pl_psgefanm.logdqok

logdquelle siehe prj.pl_psgefanm.logdquelle

logentfernt siehe prj.pl_psgefanm.logentfernt

logname siehe prj.pl_psgefanm.logname

prj_projekte_id siehe prj.pl_psgefanm.prj_projekte_id

ps_anmerkung_id siehe prj.pl_psgefanm.ps_anmerkung_id

ps_gefszenario_id siehe prj.pl_psgefanm.ps_gefszenario_id

spr_uebvar_id siehe prj.pl_psgefanm.spr_uebvar_id

Beschreibung der Sicht
Es werden die projektspezifischen (siehe prj.pl_psgefanm) Anmerkungen angezeigt.

5.20 prj.zeige__pl_psgefanm_le

Beschreibung der Spalten

prj__pl_psgefanm__id siehe prj.pl_psgefanm.id

Beschreibung der Sicht
Zu jedem in der Tabelle prj.pl_psgefanm enthaltenen Aspekt wird die jeweils letzte zugeordnete in ps verwendete Abkürzung angezeigt (Logbuchfunktion). Die unterschiedlichen Aspekte, zu denen das Logbuch geführt wird, werden anhand der Spalten prj.pl_psgefanm.prj_projekte_id, prj.pl_psgefanm.ps_gefszenario_id unterschieden.

5.21 prj.zeige__pl_psgefstrkt

Beschreibung der Spalten

id siehe prj.pl_psgefstrkt.id

notizen siehe prj.pl_psgefstrkt.notizen

logdatuhrz siehe prj.pl_psgefstrkt.logdatuhrz

logdqok siehe prj.pl_psgefstrkt.logdqok

logdquelle siehe prj.pl_psgefstrkt.logdquelle

logentfernt siehe prj.pl_psgefstrkt.logentfernt

logname siehe prj.pl_psgefstrkt.logname

prj_projekte_id siehe prj.pl_psgefstrkt.prj_projekte_id

ps_gefszenario_id siehe prj.pl_psgefstrkt.ps_gefszenario_id

str_strkt_id siehe prj.pl_psgefstrkt.str_strkt_id

Beschreibung der Sicht
Es werden die projektspezifischen Strukturpunkte angezeigt (siehe prj.pl_psgefstrkt).

5.22 prj.zeige__pl_psgefstrkt_le

Beschreibung der Spalten

prj__pl_psgefstrkt__id siehe prj.pl_psgefstrkt.id

Beschreibung der Sicht

Zu jedem in der Tabelle prj.pl_psgefstrkt enthaltenen Aspekt wird der jeweils letzte zugeordnete in ps verwendete Strukturpunkt angezeigt (Logbuchfunktion). Die unterschiedlichen Aspekte, zu denen das Logbuch geführt wird, werden anhand der Spalten prj.pl_psgefstrkt.prj_projekte_ prj.pl_psgefstrkt.ps_gefszenario_id unterschieden.

5.23 prj.zeige__pp_psgefszenario

Beschreibung der Spalten

id siehe prj.pp_psgefszenario.id

notizen siehe prj.pp_psgefszenario.notizen

prj_personen_id siehe prj.pp_psgefszenario.prj_personen_id

prj_projekte_id siehe prj.pp_psgefszenario.prj_projekte_id

ps_gefszenario_id siehe prj.pp_psgefszenario.ps_gefszenario_id

spr_uebvar_id siehe prj.pp_psgefszenario.spr_uebvar_id

spr_uebvar_id_reprgefszen siehe prj.pp_psgefszenario.spr_uebvar_id_reprgefszen

spr_uebvar_id_ursp siehe prj.pp_psgefszenario.spr_uebvar_id_ursp

Beschreibung der Sicht

Es werden die projektspezifischen Gefahrenszenarien angezeigt (siehe prj.pp_psgefszenario).

5.24 prj.zeige__ppl_dokdoku_le

Beschreibung der Spalten

prj__ppl_dokdoku__id siehe prj.ppl_dokdoku.id

Beschreibung der Sicht

Zu jedem in der Tabelle prj.ppl_dokdoku enthaltenen Dokument wird unter Berücksichtigung des Projektes die Identifikationsnummer prj.ppl_dokdoku.id des Ansprechpartners (ggf. der verantw. Person) angezeigt (Logbuchfunktion).

5.25 prj.zeige__prtprotokoll

Beschreibung der Spalten

prt__typ__id siehe prt.typ.id

prt__typ__notizen siehe prt.typ.notizen

prt__typ__aspekt siehe prt.typ.aspekt

prt__typ__spr_uebvar_id siehe prt.typ.spr_uebvar_id

prt__protokoll__id siehe prt.protokoll.id

prt__protokoll__notizen siehe prt.protokoll.notizen

prj__pp_prtprotokoll__id siehe prj.pp_prtprotokoll.id

prj__pp_prtprotokoll__notizen siehe prj.pp_prtprotokoll.notizen

prj__pp_proto koll__datuhrz siehe prj.pp_prtprotokoll.datuhrz

prj__pp_prtprotokoll__duanfrage siehe prj.pp_prtprotokoll.duanfrage

prj__pp_prtprotokoll__erledigt siehe prj.pp_prtprotokoll.erledigt

prj__pp_prtprotokoll__intern siehe prj.pp_prtprotokoll.intern

prj__pp_prtprotokoll__prj_personen_id siehe prj.pp_prtprotokoll.prj_personen_id

prj__pp_prtprotokoll__prj_projekte_id siehe prj.pp_prtprotokoll.prj_projekte_id

prj__pp_prtprotokoll__spr_uebvar_id siehe prj.pp_prtprotokoll.spr_uebvar_id

prj__pp_prtprotokoll__teilnehmer siehe prj.pp_prtprotokoll.teilnehmer

prj__pp_prtprotokoll__termin siehe prj.pp_prtprotokoll.termin

Beschreibung der Sicht

Es werden die projektspezifischen Sicherheitsanforderungen, Gefahrenprotokolleinträge und Terminplaneinträge angezeigt (siehe prj.pp_prtprotokoll).

5.26 prj.zeige__prtprotokolltabellen

Beschreibung der Spalten

prt__protokoll__id siehe prt.protokoll.id bzw. prj.p_prtverbindung.prt_protokoll_id_nachf

prj__p_prtstrstrkt__id siehe prj.p_prtstrstrkt.id

prj__p_prtstrstrkt__notizen siehe prj.p_prtstrstrkt.notizen

prj__p_prtstrstrkt__prj_projekte_id siehe prj.p_prtstrstrkt.prj_projekte_id

prj__p_prtstrstrkt__str_strkt_id siehe prj.p_prtstrstrkt.str_strkt_id

prj__p_prtverbindung__id siehe prj.p_prtverbindung.id

prj__p_prtverbindung__notizen siehe prj.p_prtverbindung.notizen

prj__p_prtverbindung__prj_projekte_id siehe prj.p_prtverbindung.prj_projekte_id

prj__p_prtverbindung__prt_protokoll_id_vorg siehe prj.p_prtverbindung.prt_protokoll_id_vorg

Beschreibung der Sicht
Es werden die projektspezifischen Zusatzinformationen (siehe prj.p_prtstrstrkt, prj.p_prtverbindung) zu den Sicherheitsanforderungen, Gefahrenprotkolleinträgen bzw. Terminplaneinträge angezeigt.

5.27 prj.zeige__psabk

Beschreibung der Spalten

ps__abk__id siehe ps.abk.id

ps__abk__notizen siehe ps.abk.notizen

ps__abk__typ siehe ps.abk.typ

prj__p_psabk__id siehe prj.p_psabk.id

prj__p_psabk__notizen siehe prj.p_psabk.notizen

prj__p_psabk__prj_projekte_id siehe prj.p_psabk.prj_projekte_id

prj__p_psabk__spr_uebvar_id siehe prj.p_psabk.spr_uebvar_id

prj__p_psabk__spr_uebvar_id_l siehe prj.p_psabk.spr_uebvar_id_l

prj__p_psabk__spr_uebvar_id_zus siehe prj.p_psabk.spr_uebvar_id_zus

Beschreibung der Sicht
Es werden die projektspezifischen (siehe prj.p_psabk) Abkürzungen angezeigt.

5.28 prj.zeige__strstrkt

Beschreibung der Spalten

str__name__id siehe str.name.id

str__name__name siehe str.name.name

str__name__name_k siehe str.name.name_k

str__name__notizen siehe str.name.notizen

str__strkt__id siehe str.strkt.id

str__strkt__notizen siehe str.strkt.notizen

str__strkt__nr siehe str.strkt.nr

prj__pp_strstrkt__id siehe prj.pp_strstrkt.id UND prj.p_strverbindung.str_strkt_id_vorg

prj__pp_strstrkt__notizen siehe prj.pp_strstrkt.notizen

prj__pp_strstrkt__fbereignis siehe prj.pp_strstrkt.fbereignis

prj__pp_strstrkt__prj_personen_id siehe prj.pp_strstrkt.prj_personen_id

prj_projekte_id siehe prj.pp_strstrkt.prj_projekte_id UND prj.p_strverbindung.prj_projekt

prj__pp_strstrkt__spr_uebvar_id siehe prj.pp_strstrkt.spr_uebvar_id

prj__p_strverbindung__id siehe prj.p_strverbindung.id

prj__p_strverbindung__notizen siehe prj.p_strverbindung.notizen

prj__p_strverbindung__fbtor siehe prj.p_strverbindung.fbtor

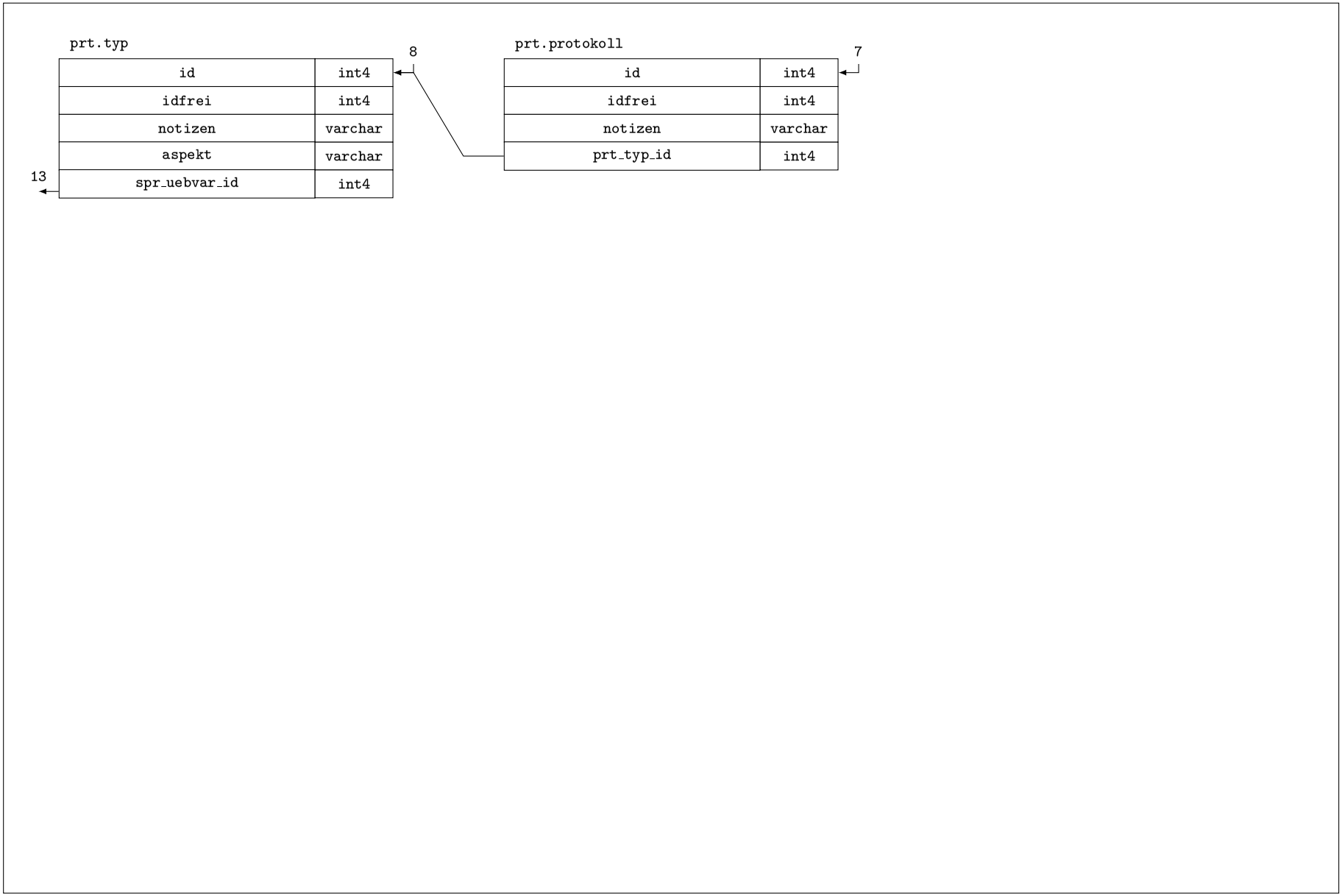
prj__p_strverbindung__str_strkt_id_nachf siehe prj.p_strverbindung.str_strkt_id_nachf

Beschreibung der Sicht
Es werden die projektübergreifend gültigen (siehe str.name, str.strkt) und die projektspezifischen Strukturpunkte (siehe prj.pp_strstrkt, prj.p_strverbindung) angezeigt.

6 prt

Im Schema „Protokoll“ werden neben Protokolleinträgen (PE) auch Sicherheitsanforderungen (SF) und Terminplaneinträge (TP) gespeichert.
Mittels der Tabelle `prt.verbindung` können Verbindungn zwischen den PE, SF, TP hergestellt werden (sind auch aspektübergreifend [PE - SF - TE] sinnvoll).

6.1 Tabellenhierarchie der Ebenen 1 — 3



6.2 Schlüssel und Bedingungen

Primärschlüssel

`pk__prt_protokoll_id` für Spalte `prt.protokoll.id`

`pk__prt_typ_id` für Spalte `prt.typ.id`

Fremdschlüssel

`fk__prt_protokoll__prt_typ_id` aus Spalte `prt.protokoll.prt_typ_id`

`fk__prt_typ__spr_uebvar_id` aus Spalte `prt.typ.spr_uebvar_id`

Unique Constraints

`uc__prt_typ` in Tabelle `prt.typ` für die Spalte(n) `prt.typ.idfrei, prt.typ.aspekt, prt.typ.spr_uebva`

Check Constraints

`cc__prt_typ` in Tabelle `prt.typ` die Bedingung `((((aspekt IS NULL) OR ((aspekt)::text = 'PE'::text)) OR ((aspekt)::text = 'SF'::text)) OR ((aspekt)::text = 'TP'::text))`

6.3 prt.nimm__protokolleintrag

Funktionsname

`prt.nimm__protokolleintrag( IN tabwerte varchar[], OUT protokollid varchar )`

Funktionsparameter/Rückgabewert(e)

`tabwerte` Liste der Werte, welche in `prt.protokoll` eingelesen werden sollen (Details siehe Funktionsbeschreibung — Werteliste einlesen). Sie werden in der folgenden Reihenfolge in die Spalten

1. `prt_typ_id` - Identifikationsnummer des Typs des Protokolleintrages.
2. `notizen`

Es müssen mindestens die ersten beiden Werte übergeben werden.

`protokollid` Es wird die Identifikationsnummer des eingelesenen Protokolleintrages zurückgegeben (`prt.protokoll.id`)

Funktionsbeschreibung

Ziel der Funktion ist das Einlesen eines Protokolleintrages, einer Sicherheitsanforderung, eines Terminplaneintrages.

Werteliste einlesen:
Dem Funktionsparameter `tabwerte` ist eine Werteliste (`ARRAY`) zu übergeben (die maximale Anzahl und die Reihenfolge der Werte ist fest vorgegeben — siehe oben). Soll nur eine Teilmenge der Werte übergeben werden, dann sind alle Werte von links nach rechts zu spezifizieren (nichtbenötigte Werte am Ende der Liste dürfen weggelassen werden).

Beispiele

```
-- Fehlermeldung. Die Mindestanzahl der Elemente muß stimmen.
SELECT * FROM prt.nimm__protokolleintrag( ARRAY['1'] );

-- Fehlermeldung. Der Parameter tabwerte[1] muß größer 0 sein.
SELECT * FROM prt.nimm__protokolleintrag( ARRAY['0', '1'] );

-- Fehlermeldung. Der Parameter tabwerte[2] muß größer 0 sein.
SELECT * FROM prt.nimm__protokolleintrag( ARRAY['1', '0'] );

-- Protokolleintrag einlesen
SELECT prt_protokoll_id, prt_typ_id, prt_protokoll__spr_uebvar_id, prt_protokoll_notizen
FROM prt.zeige__protokolleintrag;
SELECT * FROM prt.nimm__typ( ARRAY[spr.nimm__wgruppe( 'Frage', '1', '0' ), 'PE']
);
SELECT * FROM prt.nimm__protokolleintrag( ARRAY[spr.nimm__wgruppe( 'Aktueller Protokoll
'1', '0' ), '1'] );
SELECT * FROM prt.nimm__protokolleintrag( ARRAY[spr.nimm__wgruppe( 'Aktueller Protokoll
'1', '0' ), '1'] );
SELECT * FROM prt.nimm__protokolleintrag( ARRAY[spr.nimm__wgruppe( 'Weiterer Protokolle
'1', '0' ), '1', 'interne Notizen zum PE'] );
SELECT prt_protokoll_id, prt_typ_id, prt_protokoll__spr_uebvar_id, prt_protokoll_notizen
FROM prt.zeige__protokolleintrag;
```

6.4 prt.nimm__typ

Funktionsname

```
prt.nimm__typ( IN tabwerte varchar[], OUT typid varchar )
```

Funktionsparameter/Rückgabewert(e)

`tabwerte` Liste der Werte, welche in `prt.typ` eingelesen werden sollen (Details siehe Funktionsbeschreibung — Werteliste einlesen). Sie werden in der folgenden Reihenfolge in die Spalten

1. `spr_uebvar_id` - Identifikationsnummer der Wortgruppe, welche den Typ des Protokolleintrages / der Sicherheitsanforderung / den Terminplaneintrag darstellt. Sie muß größer `0` sein.
2. `aspekt` (gültige Kurzbezeichnungen siehe Funktionsbeschreibung — Aspekt - gültige Kurzbezeichnungen. Sie werden auf Plusibilität geprüft.),
3. `notizen`

Es müssen mindestens die ersten beiden Werte übergeben werden.

`typid` Es wird die Identifikationsnummer des eingelesenen Typs des Protokolleintrages / der Sicherheitsanforderung / den Terminplaneintrag zurückgegeben (`prt.typ.id`)

Funktionsbeschreibung
Ziel der Funktion ist das Einlesen eines Typs des Protokolleintrages, der Sicherheitsanforderung, des Terminplaneintrages.

Werteliste einlesen:
Dem Funktionsparameter `tabwerte` ist eine Werteliste (`ARRAY`) zu übergeben (die maximale Anzahl und die Reihenfolge der Werte ist fest vorgegeben — siehe oben). Soll nur eine Teilmenge der Werte übergeben werden, dann sind alle Werte von links nach rechts zu spezifizieren (nichtbenötigte Werte am Ende der Liste dürfen weggelassen werden).
Es wird empfohlen, zum Einlesen des ersten Parameters die Funktion `spr.nimm__wgruppe(...)` zu verwenden (Hinweise zum Umgang siehe dort).

Aspekt - gültige Kurzbezeichnungen:
Zur Klassifizierung der Aspekte sind die folgenden Kurzbezeichnungen zu verwenden.

PE Protokolleintrag

SF Sicherheitsanforderung

TP Terminplaneintrag

Beispiele

```
-- Fehlermeldung. Die Mindestanzahl der Elemente muß stimmen.
SELECT * FROM prt.nimm__typ( ARRAY['1'] );

-- Fehlermeldung. Der Parameter tabwerte[1] muß größer 0 sein.
SELECT * FROM prt.nimm__typ( ARRAY['0', 'PE'] );

-- Fehlermeldung. Der Parameter tabwerte[2] muß gültige Werte enthalten.
SELECT * FROM prt.nimm__typ( ARRAY['1', 'C'] );

-- Typ einlesen
SELECT prt_typ_id, prt_typ_notizen, prt_typ_aspekt, prt_typ__spr_uebvar_id
FROM prt.zeige__protokolleintrag;
SELECT * FROM prt.nimm__typ( ARRAY[spr.nimm__wgruppe( 'Frage', '1', '0' ), 'PE']
);
SELECT * FROM prt.nimm__typ( ARRAY[spr.nimm__wgruppe( 'Technische Maßnahme', '1',
'0' ), 'SF'] );
SELECT * FROM prt.nimm__typ( ARRAY[spr.nimm__wgruppe( 'Meilenstein', '1', '0' ),
'TP', 'interne Notizen zum TP'] );
SELECT prt_typ_id, prt_typ_notizen, prt_typ_aspekt, prt_typ__spr_uebvar_id
FROM prt.zeige__protokolleintrag;
```

7 ps

Das Schema „Produktsicherheit“ dient der Verwaltung sämtlicher Aspekte zur Produktsicherheit.

Die Zuordnung von technischen Systemen / Komponenten bzw. Strukturpunkten erfolgt **explizit**

- `str.strkt --- kmp.strkt --- kmp.komp --- ps.gefkomp --- ps.gefszenario`
- `str.strkt --- ps.gefstrkt --- ps.gefszenario`

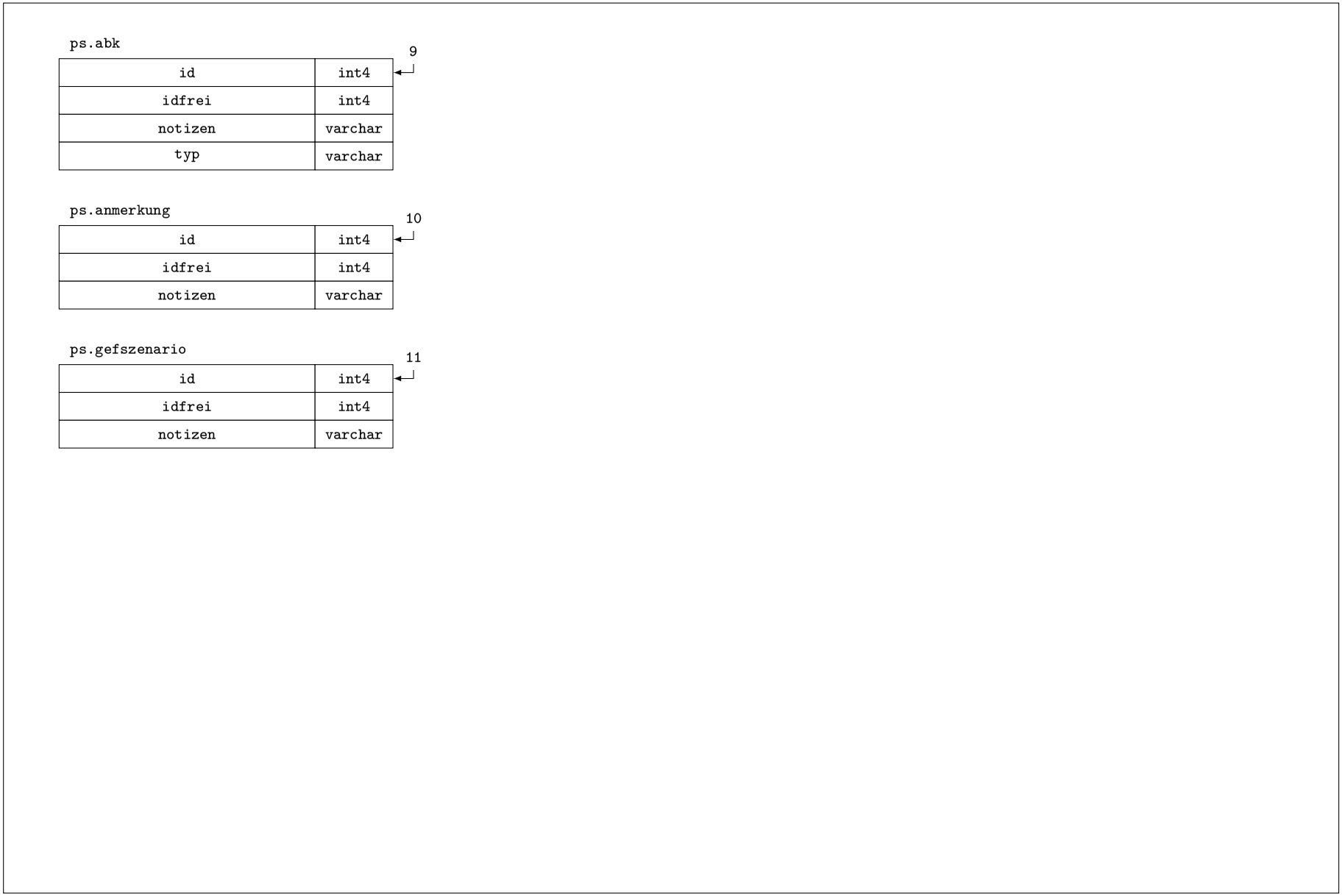
implizit über `prt`

- `str.strkt --- kmp.strkt --- kmp.komp --- prt.komp --- prt.protokoll --- ps.gefpro --- ps.gefszenario`
- `str.strkt --- prt.strkt --- prt.protokoll --- ps.gefprot --- ps.gefszenario`

implizit über `prt.dokausgabe`

- `str.strkt --- kmp.strkt --- kmp.komp --- dok.komp --- dok.doku --- dok.ausgabe --- prt.dokausgabe --- prt.protokoll --- ps.gefprot --- ps.gefszenario`
- `str.strkt --- dok.strkt --- dok.doku --- dok.ausgabe --- prt.dokausgabe --- prt.protokoll --- ps.gefprot --- ps.gefszenario`

7.1 Tabellenhierarchie der Ebenen 1 — 3



7.2 Schlüssel und Bedingungen

Primärschlüssel

`pk__ps_abk_id` für Spalte `ps.abk.id`

`pk__ps_anmerkung_id` für Spalte `ps.anmerkung.id`

`pk__ps_gefszenario_id` für Spalte `ps.gefszenario.id`

Unique Constraints

`uc__ps_abk` in Tabelle `ps.abk` für die Spalte(n) `ps.abk.idfrei, ps.abk.notizen, ps.abk.typ`

7.3 ps.nimm__abk

Funktionsname

```
ps.nimm__abk( IN tabwerte varchar[], OUT abkid varchar )
```

Funktionsparameter/Rückgabewert(e)

`tabwerte` Liste der Werte, welche in `ps.abk` eingelesen werden sollen (Details siehe Funktionsbeschreibung — Werteliste einlesen). Sie werden in der folgenden Reihenfolge in die Spalten

1. `typ` gültige Kurzbezeichnungen siehe Funktionsbeschreibung — Typ - gültige Kurzbezeichnungen. Sie werden auf Plusibilität geprüft.,
2. `notizen`

eingetragen. Es müssen beide Werte übergeben werden.

`abkid` Es wird die Identifikationsnummer der Abkürzung `ps.abk.id` zurückgegeben.

Funktionsbeschreibung

Ziel der Funktion ist das Einlesen der Abkürzungen, welche einem Gefahrenszenario zugeordnet werden können..

Werteliste einlesen:
Dem Funktionsparameter `tabwerte` ist eine Werteliste (`ARRAY`) zu übergeben (die maximale Anzahl und die Reihenfolge der Werte ist fest vorgegeben — siehe oben). Soll nur eine Teilmenge der Werte übergeben werden, dann sind alle Werte von links nach rechts zu spezifizieren (nichtbenötigte Werte am Ende der Liste dürfen weggelassen werden).

Typ - gültige Kurzbezeichnungen:

GP gefährdete Person (die dem Gefahrenszenario ausgesetzen Personen)

RG Gefahrenstufe — Auswirkungen bei Eintreten des Gefahrenszenarios (zur Risikobewertung lt. EN 50126)

RH Häufigkeit des Auftretens des Gefahrenszenarios (zur Risikobewertung lt. EN 50126)

RR Risikostufe — Bewertung des Risikos des Gefahrenszenarios (zur Risikobewertung lt. EN 50126)

Beispiele

```
-- Fehlermeldung. Die Mindestanzahl der Elemente muß stimmen.
SELECT * FROM ps.nimm__abk( ARRAY['1', '1'] );

-- Fehlermeldung. Der Parameter tabwerte[1] bzw. tabwerte[2] muß größer 0 sein.
SELECT * FROM ps.nimm__abk( ARRAY['0', '1', 'C' ] );
SELECT * FROM ps.nimm__abk( ARRAY['1', '0', 'C' ] );

-- Fehlermeldung. Der Parameter tabwerte[3] muß gültige Werte enthalten.
SELECT * FROM ps.nimm__abk( ARRAY['1', '1', 'C' ] );

-- Standardverhalten
SELECT * FROM ps.abk;
SELECT * FROM ps.nimm__abk( ARRAY[spr.nimm__wgruppe( 'abk', '1', '0' ), spr.nimm__wgruppe(
'Abkürzung', '1', '0' ), spr.nimm__wgruppe( 'Zusatzinfo', '1', '0' ), 'GP'] );
SELECT * FROM ps.nimm__abk( ARRAY[spr.nimm__wgruppe( 'abk-1', '1', '0' ), spr.nimm__wgruppe
'Abkürzung-1', '1', '0' ), spr.nimm__wgruppe( 'Zusatzinfo', '1', '0' ), 'GP', 'Notizen']
);
SELECT * FROM ps.abk;
```

7.4 ps.zeige__risikomatrix_en50126

Beschreibung der Spalten

`ps__abk__id_hfg` siehe `ps.abk.id` der Häufigkeit des Gefahrenszenarios

`ps__abk__id_gstufe` siehe `ps.abk.id` der Auswirkung des Gefahrenszenarios (Gefahrenstufe)

`ps__abk__id_rstufe` siehe `ps.abk.id` der Risikobewerttung des Gefahrenszenarios (Risikostufe)

Beschreibung der Sicht

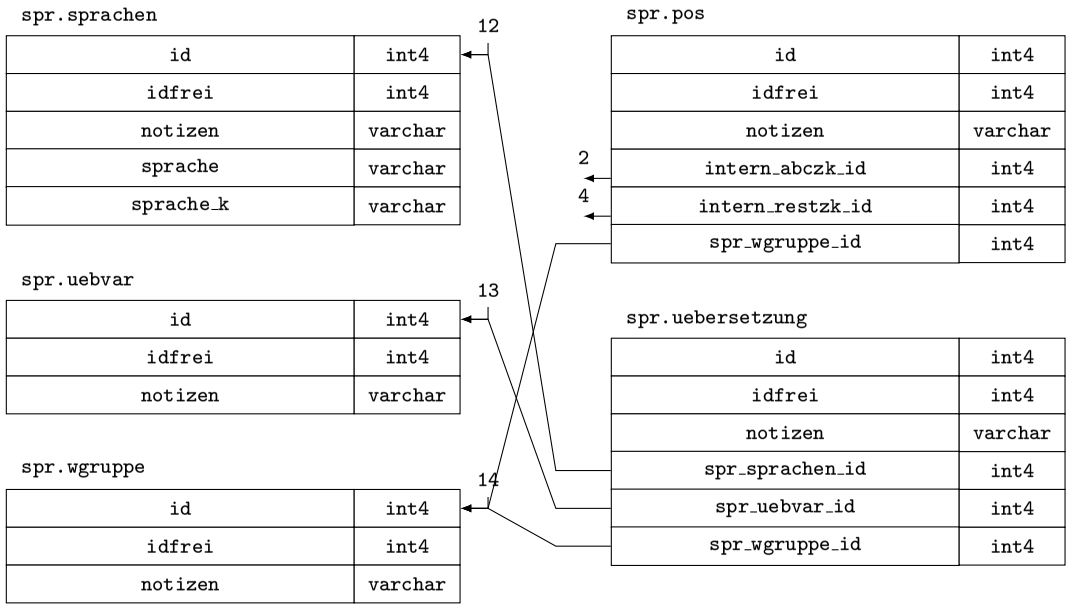
Es werden die Identifikationsnummern der Abkürzungen `ps.abk.id` zur Risikobewertung nach EN 50 126-1 ausgelesen.

8 spr

Das Schema `Sprache` dient der sprachenabhängigen Verwaltung von Wortgruppen. Jeder Wortgruppe `spr.wgruppe.id¡` können mehrere unterschiedliche Sprachen `spr.sprachen.id` mittels unterschiedlicher Übersetzungsvarianten `spr.uebvar.id` zugeordnet werden. Die Tabelle `spr.uebersetzung` dient der Zuordnung der Sprachen / Übersetzungsvarianten zur Wortgruppe.
Innerhalb einer Übersetzungsvariante `spr.uebersetzung.spr_uebvar_id` tritt jede Sprache maximal einmal auf.

Die Tabelle `spr.pos` dient der Bildung der Wortgruppen.

8.1 Tabellenhierarchie der Ebenen 1 — 3



8.2 Schlüssel und Bedingungen

Primärschlüssel

pk__spr_pos_id für Spalte spr.pos.id

pk__spr_sprachen_id für Spalte spr.sprachen.id

pk__spr_uebersetzung_id für Spalte spr.uebersetzung.id

pk__spr_uebvar_id für Spalte spr.uebvar.id

pk__spr_wgruppe_id für Spalte spr.wgruppe.id

Fremdschlüssel

fk__spr_pos_intern_abczk_id aus Spalte spr.pos.intern_abczk_id

fk__spr_pos_intern_restzk_id aus Spalte spr.pos.intern_restzk_id

fk__spr_pos_spr_wgruppe_id aus Spalte spr.pos.spr_wgruppe_id

fk__spr_uebersetzung_spr_sprachen_id aus Spalte spr.uebersetzung.spr_sprachen_id

fk__spr_uebersetzung_spr_uebvar_id aus Spalte spr.uebersetzung.spr_uebvar_id

fk__spr_uebersetzung_spr_wgruppe_id aus Spalte spr.uebersetzung.spr_wgruppe_id

Unique Constraints

uc__spr_sprachen in Tabelle spr.sprachen für die Spalte(n) spr.sprachen.idfrei, spr.sprachen.s

uc__spr_uebersetzung in Tabelle spr.uebersetzung für die Spalte(n) spr.uebersetzung.idfrei, spr.uebersetzung.spr_sprachen_id, spr.uebersetzung.spr_uebvar_id

Check Constraints

cc__spr_pos in Tabelle spr.pos die Bedingung (((((intern_abczk_id = 0) AND (intern_restzk_id = 0)) AND (spr_wgruppe_id = 0)) OR (((intern_abczk_id > 0) AND (intern_restzk_id = 0)) AND (spr_wgruppe_id > 0))) OR (((intern_abczk_id = 0) AND (intern_restzk_id > 0)) AND (spr_wgruppe_id > 0)))

8.3 spr.nimm__wgruppe

Funktionsname

```
spr.nimm__wgruppe( IN wortgruppe varchar, IN sprachenid varchar, INOUT spruebvarid
varchar )
```

Funktionsparameter/Rückgabewert(e)

wortgruppe Es ist die einzulesende Wortgruppe anzugeben. Sie darf nicht leer oder NULL sein.

`sprachenid` Es ist die Identifikationsnummer `spr.sprachen.id` der Sprache anzugeben, in der die Wortgruppe eingelesen werden soll. Der Wert muß angegeben werden und größer/gleich `0` sein.

`spruebvarid` Es wird die Identifikationsnummer (`spr.uebvar.id`) der Übersetzungsvariante zurückgegeben, welcher die aktuelle Wortgruppe einschliesslich Sprache zugeordnet wurde. Wird ein Wert größer `0` übergeben, so soll die aktuelle Wortgruppe einschliesslich Sprache dieser Übersetzungsvariante hinzugefügt werden (Modus „Übersetzen“). Andernfalls liegt der Modus „Einlesen“ vor.

Funktionsbeschreibung
Ziel der Funktion ist

- das Einlesen von Wortgruppen im Modus „Einlesen“ bzw.
- das Übersetzen einer im System vorhandenen Referenzwortgruppe `spruebvarid` mittels aktueller Wortgruppe `wortgruppe` in der aktuellen Sprache `spr.sprachen.id` im Modus „Übersetzen“.

Modus „Einlesen“
Konnte die aktuelle Wortgruppe `wortgruppe` zur gewünschten Sprache `spr.sprachen.id` im System gefunden werden, wird die erste passende Übersetzungsvariante `spr.uebvar.id` zurückgegeben. Andernfalls wird sie mit der nächsten verfügbaren `spr.uebvar.id` eingelesen.

Modus „Übersetzen“
Im System werden Übersetzungsvarianten mittels `spr.uebvar.id` gebildet. Sie stellen den konstanten Bezug auf Wortgruppen verschiedener Sprachen, welche in einem gemeinsamen Kontext stehen, dar. Innerhalb einer Übersetzungsvariante darf nur eine Übersetzung je Sprache zugeordnet werden. Somit ist es möglich unter Angabe der Sprache und Identifikationsnummer der Übersetzungsvariante die korrekte Wortgruppe zu ermitteln.
Zum korrekten Verständnis der Funktionalität im Modus „Übersetzen“ sind die folgenden Randbedingungen zu beachten:

1. Wird die aktuelle Wortgruppe zur gewünschten Sprache innerhalb der vorgegebenen Übersetzungsvariante gefunden, so wird die Identifikationsnummer der Übersetzungsvariante zurückgegeben.
2. Wird innerhalb der vorgegebenen Übersetzungsvariante keine Übersetzung in der gewünschten Sprache gefunden, wird die aktuelle Wortgruppe zur gewünschten Sprache eingelesen.
3. Wird die vorgegebenen Übersetzungsvariante nicht gefunden, so wird eine Fehlermeldung ausgegeben und das Einlesen abgebrochen.
4. Wird innerhalb der vorgegebenen Übersetzungsvariante eine Übersetzung in der gewünschten Sprache gefunden (Sprache ist bereits belegt), so wird eine Fehlermeldung ausgegeben und das Einlesen abgebrochen.

Beispiele

```
-- Fehlermeldung, die Wortgruppe ist anzugeben
SELECT * FROM spr.nimm__wgruppe( NULL, '1', NULL );
SELECT * FROM spr.nimm__wgruppe( '', '1', NULL );

-- Fehlermeldung, die Identifikationsnummer der Sprache muß größer 0 sein
SELECT * FROM spr.nimm__wgruppe( 'Das ist ein Test .', '0', NULL );

-- Fehlermeldung, die Identifikationsnummer der Übersetzungsvariante muß größer/gleich
0 sein
SELECT * FROM spr.nimm__wgruppe( 'Das ist ein Test .', '1', NULL );
SELECT * FROM spr.nimm__wgruppe( 'das ist ein Array', '1', '-1' );

-- Fehlermeldung, die Identifikationsnummer der Übersetzungsvariante muß im System
vorhanden sein
SELECT * FROM spr.nimm__wgruppe( 'das ist ein Array', '1', '999999' );

-- Fehlermeldung, die Übersetzungsvariante ist bereits mit einer Übersetzung in
der gewünschten Sprache belegt
SELECT * FROM spr.nimm__wgruppe( 'Das ist ein Array.', '1', '0' );
SELECT * FROM spr.nimm__wgruppe( 'This is an Array.', '2', '1' );
SELECT * FROM spr.zeige__wgruppe;
SELECT * FROM spr.nimm__wgruppe( '-----This is an Array.', '2', '1' );

-- Modus „Einlesen\:
-- Die aktuelle Wortgruppe konnte nicht im System gefunden werden. Sie wird neu
angelegt.
SELECT * FROM spr.zeige__wgruppe;
SELECT * FROM spr.nimm__wgruppe( 'Das ist ein Array.', '1', '0' );
SELECT * FROM spr.zeige__wgruppe;

-- Modus „Einlesen\:
-- Die aktuelle Wortgruppe konnte im System gefunden werden.
SELECT * FROM spr.nimm__wgruppe( 'Das ist ein Array.', '1', '0' );
SELECT * FROM spr.zeige__wgruppe ORDER BY spr_uebvar_id DESC LIMIT 5;
SELECT * FROM spr.nimm__wgruppe( 'Das ist ein Array.', '1', '0' );
SELECT * FROM spr.zeige__wgruppe ORDER BY spr_uebvar_id DESC LIMIT 5;

-- Modus „Übersetzen\:
-- Die aktuelle Wortgruppe konnte nicht im System gefunden werden. Die Übersetzungsvar
wird neu angelegt.
SELECT * FROM spr.nimm__wgruppe( 'Das ist ein Array.', '1', '0' );
SELECT * FROM spr.zeige__wgruppe;
SELECT * FROM spr.nimm__wgruppe( 'This is an Array.', '2', '1' );
SELECT * FROM spr.zeige__wgruppe;
```

```
-- Modus „Übersetzen\:
-- Die aktuelle Wortgruppe konnte im System gefunden werden.
SELECT * FROM spr.nimm__wgruppe( 'Das ist ein Array.', '1', '0' );
SELECT * FROM spr.nimm__wgruppe( 'This is an Array.', '2', '1' );
SELECT * FROM spr.zeige__wgruppe;
SELECT * FROM spr.nimm__wgruppe( 'This is an Array.', '2', '1' );
SELECT * FROM spr.zeige__wgruppe;
```

8.4 spr.zeige__wgelemente

Beschreibung der Spalten

`spr_uebersetzung_id` siehe `spr.uebersetzung.id`

`spr_uebvar_id` siehe `spr.uebvar.id`

`spr_sprachen_id` siehe `spr.sprachen.id`

`spr_wgruppe_id` siehe `spr.wgruppe.id`

`wgelement` siehe `intern.abczk.id ODER intern.restzk.id`

Beschreibung der Sicht

Es werden die Elemente der Wortgruppen zurückgegeben.

8.5 spr.zeige__wgruppe

Beschreibung der Spalten

`spr_uebvar_id` siehe `spr.zeige__wgelemente.spr_uebvar_id`

`spr_sprachen_id` siehe `spr.zeige__wgelemente.spr_sprachen_id`

`spr_wgruppe_id` siehe `spr.zeige__wgelemente.spr_wgruppe_id`

`wgelement` siehe `spr.zeige__wgelemente.wgelement`

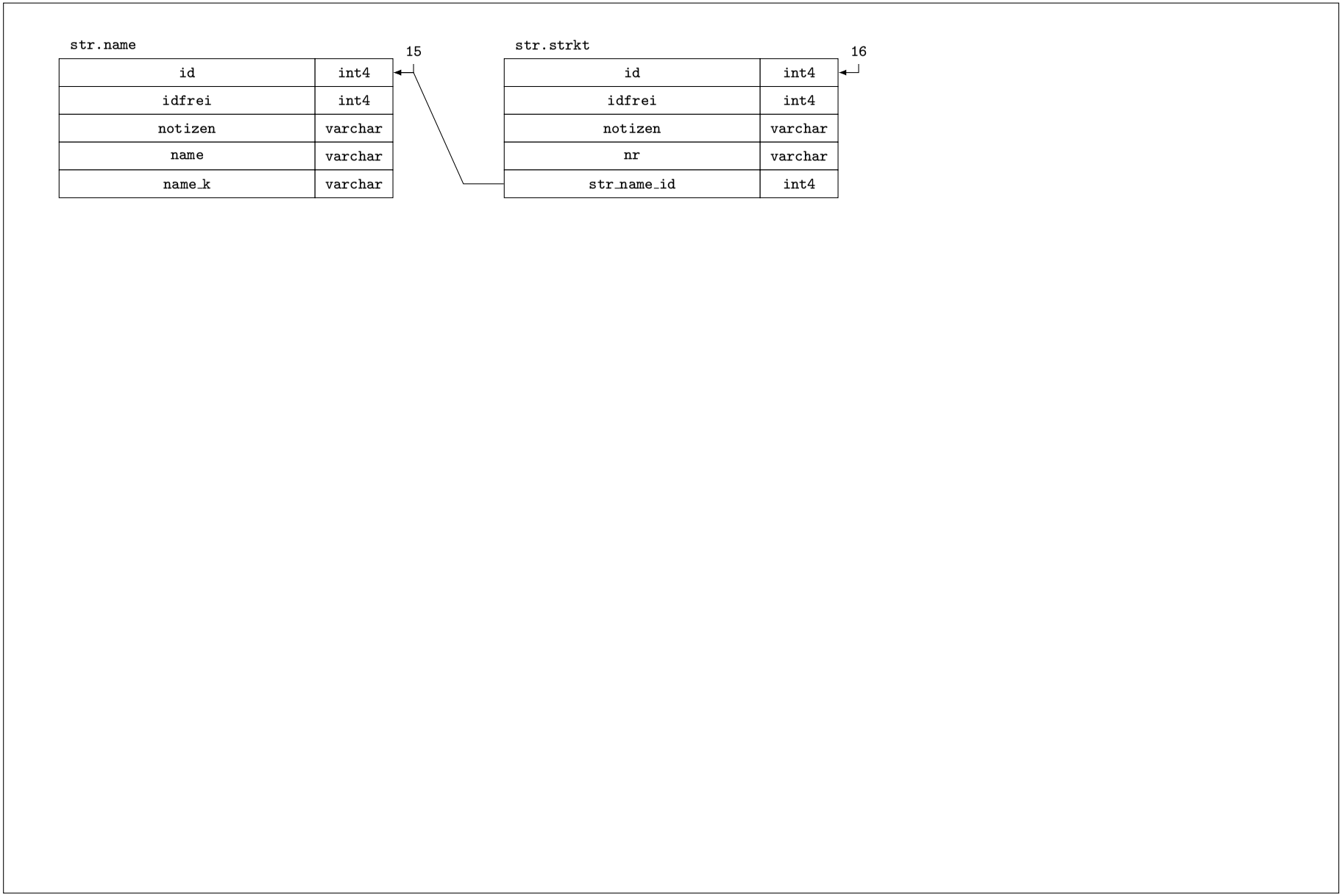
Beschreibung der Sicht

Es werden die vollständigen Wortgruppen zurückgegeben.

9 str

Das Schema „Strukturen“ dient der Verwaltung sämtlicher Strukturen (z.B. Fahrzeugstruktur, Gefahrennetz, Fehlerbaumstrukturen)

9.1 Tabellenhierarchie der Ebenen 1 — 3



9.2 Schlüssel und Bedingungen

Primärschlüssel

`pk__str_name_id` für Spalte `str.name.id`

`pk__str_strkt_id` für Spalte `str.strkt.id`

Fremdschlüssel

`fk__str_strkt__str_name_id` aus Spalte `str.strkt.str_name_id`

Unique Constraints

`uc__str_name` in Tabelle `str.name` für die Spalte(n) `str.name.idfrei, str.name.name, str.name.name_k`

`uc__str_str_strkt` in Tabelle `str.strkt` für die Spalte(n) `str.strkt.idfrei, str.strkt.nr, str.strkt.str_name_id`

9.3 str.pruefe__fbwerte

Funktionsname

`str.pruefe__fbwerte( IN aspekt varchar, IN wert varchar, IN fktname varchar, IN paramname varchar )`

Funktionsparameter/Rückgabewert(e)

`aspekt` Sollen Fehlerbaum-Ereignisse geprüft werden, ist `FB-Ereignis` anzugeben. Sollen Fehlerbaum-Tore geprüft werden, ist `FB-Tor` anzugeben.

`wert` Es ist der Wert anzugeben, dessen Plausibiliät geprüft werden soll.

`fktname` Es ist der Name der Funktion anzugeben, welche die Prüfroutine aufruft. Der Funktionsname wird in der Fehlermeldung verwendet. Sollte er leer oder `NULL` sein, wird `str.pruefe__fbwerte(...)` verwendet.

`paramname` Es ist der Name des Parameters anzugeben, welcher den zu prüfenden Wert `wert` beinhaltet. Der Parametername wird in der Fehlermeldung verwendet. Sollte er leer oder `NULL` sein, wird `wert` verwendet.

Funktionsbeschreibung

Ziel der Funktion ist die Plausibilitätsprüfung gültiger Fehlerbaum-Ereignisse bzw. -Tore.

`FB-Ereignis` - gültige Werte nach DIN IEC 61025:1993-12):

EBB Ereignis-Beschreibungs-Block

GE Grundereignis

NUE Nicht weiter untersuchtes Ereignis

AUE Anderweitig untersuchtes Ereignis

HAUS Haus

NULL Null-Ereignis

VWVA Verweisungs-Zeichen von außen

VWNA Verweisungs-Zeichen nach außen

`FB-Tor` - gültige Werte nach DIN IEC 61025:1993-12):

UND UND-Verknüpfung

ODER ODER-Verknüpfung

XODER EXCLUSIV-ODER-Verknüpfung

NICHT NICHT-Verknüpfung

BED Bedingungs-Verknüpfung

RED Redundante Struktur

ALLG Verknüpfung (allgemein)

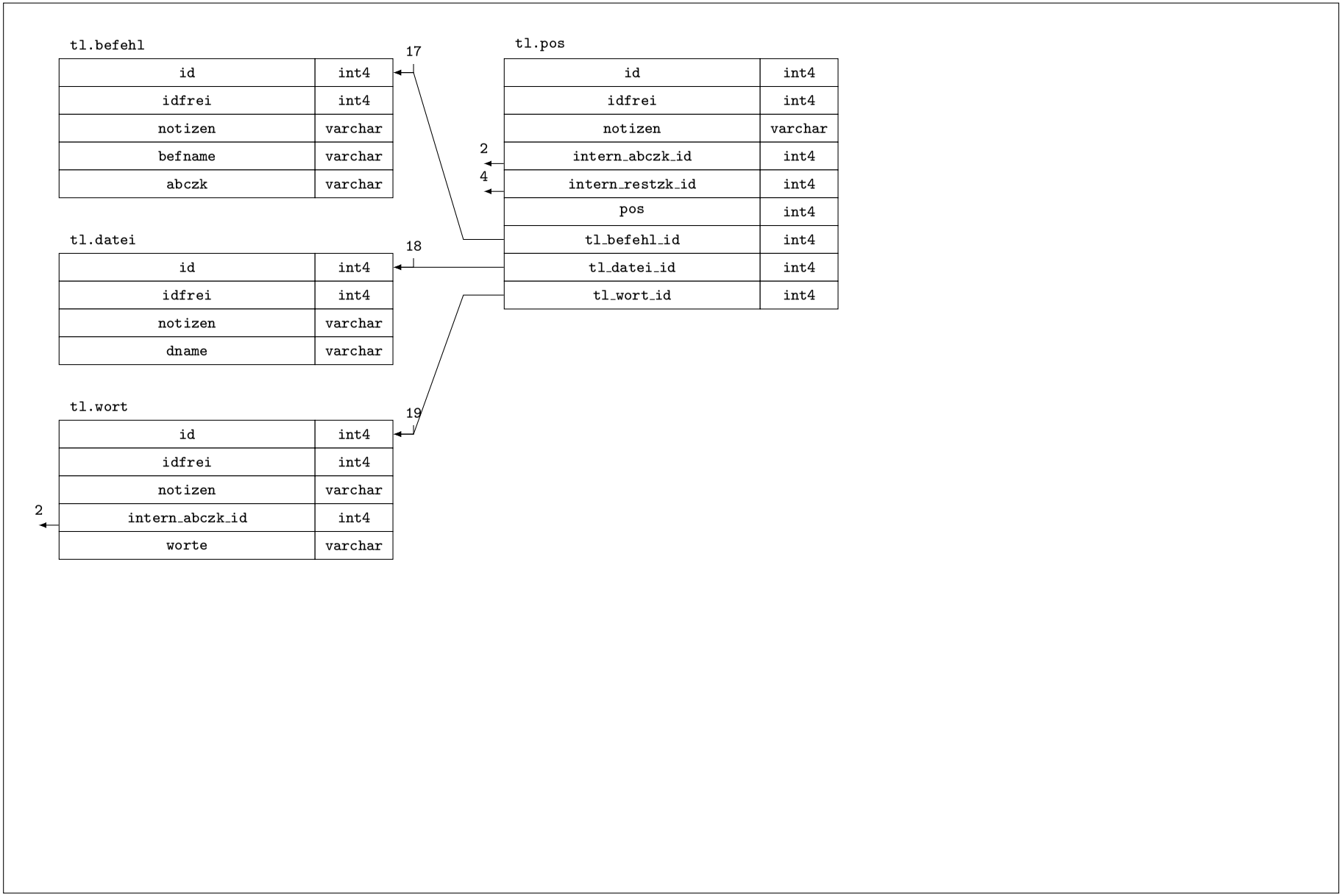
Beispiele

```
-- Fehlermeldung. Der übergebene Aspekt ist ungültig.
SELECT * FROM str.pruefe__fbwerte( NULL, 'HAUS', NULL, NULL );

-- Fehlermeldung. Der übergebene Wert darf Nicht NULL oder leer sein, bzw. muß
dem gültigen Wert entsprechen
SELECT * FROM str.pruefe__fbwerte( 'FB-Ereignis', NULL, NULL, NULL );
SELECT * FROM str.pruefe__fbwerte( 'FB-Ereignis', 'ungueltig', NULL, NULL );
SELECT * FROM str.pruefe__fbwerte( 'FB-Tor', 'ungueltig', 'fktname', 'fktparam'
);

-- Die Werte sind gültig
SELECT * FROM str.pruefe__fbwerte( 'FB-Ereignis', 'HAUS', NULL, NULL );
SELECT * FROM str.pruefe__fbwerte( 'FB-Tor', 'UND', NULL, NULL );
```

10 tl

Das Schema „Tex/Latex“ dient der Verwaltung der verwendeten TEX/LATEX - Befehle, TEX/LATEX - Wörter und sonstigen in TEX/LATEX - Dateien auftretenden Zeichenketten.

10.1 Tabellenhierarchie der Ebenen 1 — 3



10.2 Schlüssel und Bedingungen

Primärschlüssel

pk__tl_befehl_id für Spalte tl.befehl.id

pk__tl_datei_id für Spalte tl.datei.id

pk__tl_pos_id für Spalte tl.pos.id

pk__tl_wort_id für Spalte tl.wort.id

Fremdschlüssel

fk__tl_pos__intern_abczk_id aus Spalte tl.pos.intern_abczk_id

fk__tl_pos__intern_restzk_id aus Spalte tl.pos.intern_restzk_id

fk__tl_pos__tl_befehl_id aus Spalte tl.pos.tl_befehl_id

fk__tl_pos__tl_datei_id aus Spalte tl.pos.tl_datei_id

fk__tl_pos__tl_wort_id aus Spalte tl.pos.tl_wort_id

fk__tl_wort__intern_abczk_id aus Spalte tl.wort.intern_abczk_id

Unique Constraints

uc__tl_befehl in Tabelle tl.befehl für die Spalte(n) tl.befehl.idfrei, tl.befehl.befname, tl.befehl.abczk

uc__tl_datei in Tabelle tl.datei für die Spalte(n) tl.datei.idfrei, tl.datei.dname

uc__tl_pos in Tabelle tl.pos für die Spalte(n) tl.pos.idfrei, tl.pos.pos, tl.pos.tl_datei_id

uc__tl_wort in Tabelle tl.wort für die Spalte(n) tl.wort.idfrei, tl.wort.intern_abczk_id, tl.wort.worte

Check Constraints

cc__tl_pos in Tabelle tl.pos die Bedingung

```
(((((((intern_abczk_id = 0) AND (intern_restzk_id
= 0)) AND (tl_befehl_id = 0)) AND (tl_wort_id = 0)) OR ((((intern_abczk_id >
0) AND (intern_restzk_id = 0)) AND (tl_befehl_id = 0)) AND (tl_wort_id = 0)))
OR ((((intern_abczk_id = 0) AND (intern_restzk_id > 0)) AND (tl_befehl_id =
0)) AND (tl_wort_id = 0))) OR ((((intern_abczk_id = 0) AND (intern_restzk_id
= 0)) AND (tl_befehl_id > 0)) AND (tl_wort_id = 0))) OR ((((intern_abczk_id
= 0) AND (intern_restzk_id = 0)) AND (tl_befehl_id = 0)) AND (tl_wort_id > 0)))
```

10.3 tl.erzeuge__abczk

Funktionsname
`tl.erzeuge__abczk( IN tlwort varchar, OUT abczk varchar )`

Funktionsparameter/Rückgabewert(e)

`tlwort` Es ist das Wort anzugeben, aus welchem die AbCZk extrahiert werden soll. ACHTUNG ! Das '\' ist doppelt anzugeben (zu maskieren). Bsp.: `tl.erzeuge__ABCZk( 'Änderung' )`

`abczk` Es wird die aus dem TL-Wort erzeugte AbCZk zurückgegeben.

Funktionsbeschreibung
Ziel der Funktion ist das Erzeugen einer AbCZk `abcZk` aus dem übergebenen TL-Wort `tlwort`. Es werden alle TL-Zweizeichenbefehle in `tlwort` mittels der in `tl.befehl.abczk` zugeordneten AbCZk ersetzt.
Sollte `tlwort` leer sein, oder beim Erzeugen ein Fehler auftreten bzw. `abcZk` kein AbCZk darstellen, wird eine Fehlermeldung generiert.

Beispiele
```
-- Fehlermeldung. Der Parameter tlwort darf nicht leer oder NULL sein.
SELECT * FROM tl.erzeuge__ABCZk( '' );

-- Fehlermeldung. Es konnten keine TL-Zweizeichenbefehle gefunden werden.
SELECT * FROM tl.erzeuge__ABCZk( 'Änderung' );

-- Änderung zurückgeben
SELECT intern.nimm__zeile( 'tl.befehl', ARRAY['abczk', 'befname'], ARRAY['Ä', 'Ä']
);
SELECT intern.nimm__zeile( 'tl.befehl', ARRAY['befname'], ARRAY[E''] );
SELECT * FROM tl.erzeuge__ABCZk( E'Änderung' );
```

10.4 tl.erzeuge__dbmodell

Funktionsname
`tl.erzeuge__dbmodell( OUT tldateiinhalt varchar )`

Funktionsparameter/Rückgabewert(e)

`tldateiinhalt` Es wird der Inhalt der TEX/LATEX-Datei zurückgegeben, welche das Datenbankmodell enthält.

Funktionsbeschreibung
Ziel der Funktion ist das Erzeugen einer TEX/LATEX-Datei, welche das Datenbankmodell enthält. Sie wird aus den Kommentaren erzeugt, welche den verschiedenen Datenbankobjekten zugeordnet wurden. Es werden die Kommentare der Datenbank, Schemas, Tabellen,

Spalten, Funktionen, Primär- und Fremdschlüssel, sowie die `unique` und `check constraints` verwendet.
In den Kommentaren sind \ entsprechend zu doppeln. Unterstrichen werden automatisch \ vorangestellt, weshalb zum momentanen Zeitpunkt keine mathematischen Formeln eingegeben werden können.

Hinweise zur Benutzung:
Um nicht auf eine separate Ausleseroutine angewiesen zu sein, wird folgende (in `psql`) auszufuehrende Handlungsweise empfohlen:

```
\pset tuples_only \pset format unaligned \pset footer
SELECT * FROM tl.erzeuge__DBModell() \o dateiname.tex
;
\q
```

Alternativ kann in der Kommandozeile

```
psql -U micha vf -P tuples_only -P format=unaligned -P footer -c ‘‘SELECT * FROM tl.erzeuge__DBModell()‘‘ -o dateiname.tex
```

eingegeben werden.

Beispiele

```
-- Datenbankmodell ausgeben
SELECT * FROM tl.erzeuge__DBModell();
```

10.5 tl.erzeuge__tldbglossary

Funktionsname

```
tl.erzeuge__tldbglossary( IN begriff varchar, IN gloprfx varchar, IN bechreibung varchar, OUT tldbgl varchar )
```

Funktionsparameter/Rückgabewert(e)

`begriff` Der im Textdokument und Glossar darzustellende Begriff (darf nicht leer sein).

`gloprfx` Im Glossar wird `gloPrfx` dem `begriff` vorangestellt (zur Positionierung im Glossar) - darf leer sein.

`bechreibung` Die im Glossar dem `begriff` zuzuordnende Beschreibung (darf leer sein).

`tldbgl` Der im TL-Dokument einzusetzende `\dbGlossary...` - Befehl.

Funktionsbeschreibung
Ziel der Funktion ist das Erzeugen eines `\dbGlossary...` - Befehls welcher im TL-Dokument verwendet werden kann.
Sollte `bechreibung` leer sein, wird ausschließlich `begriff` zurückgegeben.

Beispiele

```
-- Fehlermeldung. Der Parameter begriff darf nicht leer oder NULL sein.
SELECT * FROM tl.erzeuge__TLdbGlossary( NULL, '', '' );
SELECT * FROM tl.erzeuge__TLdbGlossary( '', '', '' );

-- Glossareintrag zurückgeben
SELECT * FROM tl.erzeuge__TLdbGlossary( 'cc__intern_alle', 'intern.alle', NULL );
SELECT * FROM tl.erzeuge__TLdbGlossary( 'cc__intern_alle', 'intern.alle', '' );
SELECT * FROM tl.erzeuge__TLdbGlossary( 'cc__intern_alle', 'intern.alle', 'Es sind
die Zeichenketten einzugeben.' );
```

10.6 tl.erzeuge__tlzk

Funktionsname
`tl.erzeuge__tlzk( IN zk varchar, OUT tlzk varchar )`

Funktionsparameter/Rückgabewert(e)

`zk` Es ist die in TEX/LATEX zu wandelnde Zeichenkette anzugeben.

`tlzk` Es wird die in TEX/LATEX gewandelte Zeichenkette zurückgegeben.

Funktionsbeschreibung
Ziel der Funktion ist das Umwandeln von Zeichenketen in TEX/LATEX-Zeichenketten, welche im Absatzmodus akzeptiert werden können.

Beispiele
```
-- Es wird eine Leerzeichenkette bzw. NULL zurückgegeben
SELECT * FROM tl.erzeuge__TLZk( '' );
SELECT * FROM tl.erzeuge__TLZk( NULL );

-- Es wird intern_abczk_id zurückgegeben
SELECT * FROM tl.erzeuge__TLZk( 'intern_abczk_id' );

-- Es wird normale Zk zurückgegeben
SELECT * FROM tl.erzeuge__TLZk( 'normale Zk' );
```

10.7 tl.gib__datei

Funktionsname
`tl.gib__datei( IN dateiname varchar, OUT dateiinhalt varchar )`

Funktionsparameter/Rückgabewert(e)

`dateiname` Vollständiger Dateiname (einschl. Dateierweiterung) der auszugebenden TL-Datei

`dateiinhalt` Vollständiger Inhalt der auszugebenden TL-Datei

Funktionsbeschreibung
Ziel der Funktion ist das Herausgeben des Inhalts der mittels `dateiname` spezifizierten TEX/LATEX-Quelltextdatei (TL-Datei).

Hinweise zur Benutzung:
Um nicht auf eine separate Ausleseroutine angewiesen zu sein, wird folgende (in `psql`) auszufuehrende Handlungsweise empfohlen:

```
\pset tuples_only
\pset format unaligned
\pset footer
SELECT * FROM tl.gib__datei( 'dateiname.tex' ) \o dateiname.tex
;
\q
```

Alternativ kann in der Kommandozeile
`psql -U micha vf -P tuples_only -P format=unaligned -P footer -c ``SELECT * FROM tl.gib__datei( 'dateiname.tex' );`` -o dateiname.tex`
eingegeben werden.

Beispiele

```
-- Fehlermeldung. Der Parameter dateiname darf nicht leer oder NULL sein.
SELECT * FROM tl.gib__datei( NULL );

-- Dateiinhalt einlesen und wieder ausgeben
SELECT intern.nimm__zeile( 'tl.befehl', ARRAY['abczk', 'befname'], ARRAY['Ä', '
Ä'] );
SELECT intern.nimm__zeile( 'tl.befehl', ARRAY['befname'], ARRAY[E'\-'] );
SELECT intern.nimm__zeile( 'tl.datei', ARRAY['dname'], ARRAY['dateiname.tex'] );
SELECT * FROM tl.nimm__datei(
E'Än\-de\-rung
Än\-de\-rungHo\-sen\-bein
'' 2006-01-01 #1 Änderung
norm-dt_01050_din_en
\befehl'
, 'dateiname.tex' );
SELECT * FROM tl.gib__datei( 'dateiname.tex' );
```

10.8 tl.gib__ere_tlbefehl

Funktionsname
`tl.gib__ere_tlbefehl( OUT ere_tlbefehl varchar )`

Funktionsparameter/Rückgabewert(e)

`ere_tlbefehl` Ist der erweiterte regulaere Ausdruck zur Beschreibung eines TL-Befehls

Funktionsbeschreibung
Ziel der Funktion ist das Definieren und Zurueckgeben eines erweiterten regulaeren Ausdruckes (ERE), welcher das Textmuster eines TEX/LATEX - Befehls (TL-Befehl) beschreibt.
TL-Befehle beginnen mit einem oder @ oder Buchstaben dem ein Buchstabe folgen muß. TL-Befehle, welche mit einem Buchstaben beginnen, müssen mindestens ein - _ oder eine Ziffer enthalten.

TL-Zweizeichenbefehle werden über `tl.gib__ERE_TLZZBefehl()` abgefangen.
TL-Befehle, welche mit - _ oder einer Ziffer beginnen, werden bis zum ersten Buchstaben `intern.gib__ERE_RestZK()` abgebildet (TL-Befehl wird zerteilt).
Klammern und Sterne welche Teil eines TL-Befehls sein können, werden mittels `intern.gib__ERE_RestZK()` abgebildet (TL-Befehl wird zerteilt).

Es ist besonders zu beachten, daß auch ABCZk (siehe `intern.gib__ERE_ABCZK()`) beschrieben werden, solange sie keine deutschen Sonderzeichen (ÄÖÜ, ...) enthalten ! Das ist von folgenden Routinen abzufangen !

Beispiele

```
-- ERE zurückgeben
SELECT * FROM tl.gib__ERE_TLBefehl();

-- Es wird test zurückgegeben
SELECT substring( 'test' from tl.gib__ERE_TLBefehl() );

-- Es wird \befehl zurückgeben
SELECT substring( '\befehl' from tl.gib__ERE_TLBefehl() );

-- Es wird @manual zurückgeben
SELECT substring( '@manual' from tl.gib__ERE_TLBefehl() );

-- Es wird abs2def zurückgeben
SELECT substring( 'abs2def' from tl.gib__ERE_TLBefehl() );

-- Es wird absdef2 zurückgeben
SELECT substring( 'absdef2' from tl.gib__ERE_TLBefehl() );

-- Es wird norm-dt_01050_din_en zurückgeben
SELECT substring( 'norm-dt_01050_din_en' from tl.gib__ERE_TLBefehl() );

-- Es wird eine Leerzeichenkette zurückgeben
SELECT substring( '\' from tl.gib__ERE_TLBefehl() );
```

```
-- Es wird eine Leerzeichenkette zurückgeben
SELECT substring( '\\' from tl.gib__ERE_TLBefehl() );

-- Es wird eine Leerzeichenkette zurückgeben
SELECT substring( '\@befehl' from tl.gib__ERE_TLBefehl() );

-- Es wird \bef zurückgeben (leider)
SELECT substring( '\bef@befehl' from tl.gib__ERE_TLBefehl() );

-- Es wird eine Leerzeichenkette zurückgeben
SELECT substring( '2absdef' from tl.gib__ERE_TLBefehl() );

-- Es wird eine Leerzeichenkette zurückgeben
SELECT substring( '-no-dt_01050_din_en' from tl.gib__ERE_TLBefehl() );

-- Es wird eine Leerzeichenkette zurückgeben
SELECT substring( '_no-dt_01050_din_en' from tl.gib__ERE_TLBefehl() );

-- Es wird eine Leerzeichenkette zurückgeben
SELECT substring( '2no-dt_01050_din_en' from tl.gib__ERE_TLBefehl() );
```

10.9 tl.gib__ere_tlwort

Funktionsname

`tl.gib__ere_tlwort( OUT ere_tlwort varchar )`

Funktionsparameter/Rückgabewert(e)

`ere_tlwort` Ist der erweiterte reguläre Ausdruck zur Beschreibung eines TL-Wortes.

Funktionsbeschreibung

Ziel der Funktion ist das Definieren und Zurückgeben eines erweiterten regulären Ausdruckes (ERE), welcher das Textmuster eines TL-Wortes beschreibt.
Ein TL-Wort beginnt mit einem Buchstaben oder mit einem ïn Kombination mit dem zulässigen Buchstaben.
In der Folge setzt es sich aus TL-Zweizeichenbefehlen (siehe `tl.gib__ERE_TLZZBefehl()`) und ABCZk (siehe `intern.gib__ERE_ABCZK()`) zusammen.
Es ist besonders zu beachten, daß auche ASCII-Wörter (siehe `intern.gib__ERE_ABCZK()`) beschrieben werden, solange sie keine deutschen Sonderzeichen (ÄÖÜ, ...) enthalten ! Das ist von folgenden Routinen abzufangen !

Beispiele

```
-- ERE zurückgeben
SELECT * FROM tl.gib__ERE_TLWort();
```

```
-- Es wird test zurückgegeben (leider)
SELECT substring( 'test' from tl.gib__ERE_TLWort() );

-- Es wird eine Leerzeichenkette zurückgegeben
SELECT substring( E'\Aistfalsch' from tl.gib__ERE_TLWort() );

-- Es wird Äßn-de-rung zurückgegeben
SELECT substring( E'Äßnderung' from tl.gib__ERE_TLWort() );

-- Es wird Äßnde-rungä zurückgegeben
SELECT substring( E'Äßnderungä' from tl.gib__ERE_TLWort() );

-- Es wird ßnderungä zurückgegeben (leider)
SELECT substring( E'ßnderungä' from tl.gib__ERE_TLWort() );

-- Es wird Äßmann zurückgegeben
SELECT substring( 'Äßmann' from tl.gib__ERE_TLWort() );
```

10.10 tl.gib__ere_tlzzbefehl

Funktionsname
`tl.gib__ere_tlzzbefehl( IN istanfang bool, OUT ere_tlzzbefehl varchar )`

Funktionsparameter/Rückgabewert(e)

`istanfang` Der erweiterte reguläre Ausdruck des Anfangs eines TL-Zweizeichenbefehls soll zurückgegeben werden

`ere_tlzzbefehl` Ist der erweiterte reguläre Ausdruck zur Beschreibung eines TL-Zweizeichenbefehls.

Funktionsbeschreibung
Ziel der Funktion ist das Definieren und Zurückgeben eines erweiterten regulären Ausdruckes (ERE), welcher das Textmuster eines TL-Zweizeichenbefehls beschreibt. Die folgenden TL-Zweizeichenbefehle werden vom ERE erfaßt:

- Ä
- ä
- Ö
- ö
- Ü
- ü
-

-

Beispiele

```
-- ERE zurückgeben
SELECT * FROM tl.gib__ERE_TLZZBefehl( true );
SELECT * FROM tl.gib__ERE_TLZZBefehl( false );

-- Leerzeichenkette zurückgeben
SELECT substring( E'' from tl.gib__ERE_TLZZBefehl( true ) );

-- ü zurückgeben
SELECT substring( 'ü' from tl.gib__ERE_TLZZBefehl( true ) );

--  zurückgeben
SELECT substring( E'' from tl.gib__ERE_TLZZBefehl( false ) );
```

10.11 tl.gib__liste_tlzzbefehl

Funktionsname

```
tl.gib__liste_tlzzbefehl( IN tlwort varchar, OUT listetlzzbefehl varchar[] )
```

Funktionsparameter/Rückgabewert(e)

`tlwort` Es ist das Wort anzugeben, aus welchem die TL-Zweizeichenbefehle extrahiert werden sollen. ACHTUNG ! Das '\' ist doppelt anzugeben (zu maskieren). Bsp.: `tl.gib__Liste_TLZZBefeh 'Änderung' )`

`listetlzzbefehl` Es wird die Liste der gefundenen TL-Zweizeichenbefehle als `ARRAY` vom Typ `VARCHAR` zurückgegeben. Sollte kein TL-Zweizeichenbefehl in `tlwort` enthalten oder `tlwort` leer sein, wird `NULL` zurückgegeben.

Funktionsbeschreibung

Ziel der Funktion ist das Extrahieren von TL-Zweizeichenbefehlen aus einem Wort. Das Wort besteht ausschließlich aus Buchstaben und (ggf.) aus TL-Zweizeichenbefehlen. Es wird nicht geprüft, ob die extrahierten TL-Zweizeichenbefehle im System vorhanden sind.

Beispiele

```
-- Es wird NULL zurückgegeben
SELECT * FROM tl.gib__Liste_TLZZBefehl( NULL );

-- Es wird NULL zurückgegeben
SELECT * FROM tl.gib__Liste_TLZZBefehl( '' );

-- Es wird NULL zurückgegeben
```

```
SELECT * FROM tl.gib__Liste_TLZZBefehl( 'Änderung' );

-- Es wird Ä zurückgegeben
SELECT * FROM tl.gib__Liste_TLZZBefehl( 'Änderung' );

-- Es wird Ä  zurückgegeben
SELECT * FROM tl.gib__Liste_TLZZBefehl( E'Änderung' );

-- Es wird  zurückgegeben
SELECT * FROM tl.gib__Liste_TLZZBefehl( E'Änderung' );
```

10.12 tl.ist__tlwort

Funktionsname
`tl.ist__tlwort( IN tlwort varchar, OUT isttlwort bool )`

Funktionsparameter/Rückgabewert(e)

`tlwort` Es ist das zu testende Wort anzugeben. ACHTUNG ! Das '\' ist doppelt anzugeben (zu maskieren). Bsp.: `tl.ist__TLWort( 'Änderung' )`

`isttlwort` Wenn `istTLWort = false` dann ist `tlwort` kein TL-Wort. Wenn `istTLWort = true` dann ist ein TL-Wort

Funktionsbeschreibung
Ziel der Funktion ist das Prüfen, ob die übergebene Zeichenkette `tlwort` als TL-Wort interpretiert werden kann.
TL-Worte bestehen ausschließlich aus AbcZk in Kombination mit mindestens einem TL-Zweizeichenbefehl (Definition und Gültigkeit siehe `tl.gib__ERE_TLWort()` unter der Bedingung, daß `intern.ist__abczk( tlwort ) = false` ist).

Beispiele
```
-- Es wird false zurückgegeben
SELECT * FROM tl.ist__TLWort( '' );

-- Es wird false zurückgegeben
SELECT * FROM tl.ist__TLWort( NULL );

-- Es wird false zurückgegeben
SELECT * FROM tl.ist__TLWort( 'Hosenbein' );

-- Es wird false zurückgegeben
SELECT * FROM tl.ist__TLWort( 'Änderung__' );

-- Es wird true zurückgegeben
```

```
SELECT * FROM tl.ist__TLWort( 'Änderung' );

-- Es wird true zurückgegeben
SELECT * FROM tl.ist__TLWort( 'Änderung' );

-- Es wird true zurückgegeben
SELECT * FROM tl.ist__TLWort( 'Hosenbein' );
```

10.13 tl.nimm__datei

Funktionsname
`tl.nimm__datei( IN dateiinhalt varchar, IN dateiname varchar, OUT tldateiid varchar )`

Funktionsparameter/Rückgabewert(e)

`dateiinhalt` Vollständiger Inhalt der einzulesenden TL-Datei (auch Zusatzmodule wie `bibtex`)

`dateiname` Vollständiger Dateiname (einschl. Dateierweiterung) der einzulesenden TL-Datei

`tldateiid` Es wird die Identifikationsnummer des Namens der Datei zurückgegeben, deren Inhalt eingelesen wurde.

Funktionsbeschreibung
Ziel der Funktion ist das Einlesen des Inhalts der mittels `dateiname` spezifizierten TEX/LATEX-Quelltextdatei (TL-Datei). Auch Quelltextdateien von TL-Zusatzpaketen (z.B. `bibtex`) werden als TL-Dateien interpretiert. Die Funktion extrahiert aus dem übergebenen TL-Dateiinhalt die enthaltenen TL-Objekte (siehe `tl.nimm_erkannte__Liste_TLObjekte(...)`) und liest sie in `tl.pos` ein.
Das Konzept zum extrahieren der TL-Objekte kann der Beschreibung zu `tl.nimm_erkannte__Liste_TLOb` entnommen werden

Hinweise zur Benutzung:
Um nicht auf eine separate Einleseroutine angewiesen zu sein, wird folgende (in `psql` auszuführende Handlungsweise empfohlen):

1. Es ist sichergestellt, daß die einzulesende TL-Datei im TEX/LATEX fehlerfrei compiliert werden kann

2. Es ist eine Kopie der einzulesenen TL-Datei anzulegen

3. In der Kopie sind alle \
   mittels \\
   und alle '
   mittels ''
   zu ersetzen

4. Dem ersten Zeichen in der Kopie ist der SQL-Teilbefehl
   `SELECT * FROM tl.nimm__datei( E'`
   voranzustellen
5. Dem letzten Zeichen in der Kopie mit dem Dateinamen `dateiname.tex` ist der SQL-Teilbefehl
   `', 'dateiname.tex');`
   hizuzufügen
6. Im Postgres-Kommandozeilenprogramm `psql` ist mittels des Befehls
   `\i dateiname.tex`
   die in den beschriebenen Schritten modifizierte Kopie einzulesen

Beispiele

```
-- Fehlermeldung. Der Parameter dateiinhalt darf nicht leer oder NULL sein.
SELECT * FROM tl.nimm__datei( '', 'dateiname.tex' );

-- Fehlermeldung. Der Parameter dateiname darf nicht leer oder NULL sein.
SELECT * FROM tl.nimm__datei( 'ttt', NULL );

-- Fehlermeldung. Der Dateiname muß im System vorhanden sein.
SELECT * FROM tl.nimm__datei( 'ttt', 'Unbekannter Dateiname.tex' );

-- Dateiinhalt einlesen und wieder ausgeben
SELECT intern.nimm__zeile( 'tl.befehl', ARRAY['abczk', 'befname'], ARRAY['Ä', '
Ä'] );
SELECT intern.nimm__zeile( 'tl.befehl', ARRAY['befname'], ARRAY[E'\-'] );
SELECT intern.nimm__zeile( 'tl.datei', ARRAY['dname'], ARRAY['dateiname.tex'] );
SELECT * FROM tl.nimm__datei(
E'Än\-de\-rung
Än\-de\-rungHo\-sen\-bein
'' 2006-01-01 #1 Änderung
norm-dt_01050_din_en
\befehl'
, 'dateiname.tex' );
SELECT * FROM tl.gib__datei( 'dateiname.tex' );
```

10.14 tl.nimm__tlwort

Funktionsname

```
tl.nimm__tlwort( IN tlwort varchar, IN tlwortid_korr varchar, OUT tlwortid varchar
)
```

Funktionsparameter/Rückgabewert(e)

`tlwort` Einzulesendes TL-Wort. ACHTUNG ! Das '\' ist doppelt anzugeben (zu maskieren).
Bsp.: `tl.nimm__TLWort( 'Ho`
`-sen`
`-bein', true )`

`tlwortid_korr` Wird eine im System vorhandene `tl.wort.id` benannt, so wird das im System vorhandene TL-Wort überschrieben. Soll ein Wort hinzugefügt werden, so ist entweder '0' oder `NULL` anzugeben.

`tlwortid` Es wird die Identifikationsnummer des eingelesenen TL-Wortes zurueckgegeben.

Funktionsbeschreibung
Ziel der Funktion ist das Einlesen des übergebenen TL-Wortes `tlWort` in die Tabelle `tl.wort`. Das gültige Textmuster eines TL-Wortes kann der Funktion `tl.ist__TLWort(...)` entnommen werden.
Die im TL-Wort enthaltenen TL-Zweizeichenbefehle werden extrahiert, deren AbcZk-Bedeutung aus `tl.befehl.abczk` ausgelesen und im TL-Wort an Stelle der TL-Zweizeichenbefehle eingesetzt. Die auf diese Weise erzeugte AbcZk wird in `intern.abczk` eingelesen und mit dem in `tl.wort` eingelesenen TL-Wort verknüpft.
Wird eine mittels `tlWortId_korr` übergebene `tl.wort.id` im System gefunden, so wird das im System vorhandene TL-Wort überschrieben. Andernfalls wird `tlWort` mit der vorgegebenen `tlWortId_korr` im System eingefügt.

Beispiele
```
-- Fehlermeldung. Es liegt kein TL-Wort vor
SELECT * FROM tl.nimm__TLWort( '', NULL );
SELECT * FROM tl.nimm__TLWort( 'Hosenbain', NULL ); -- Fehlermeldung

-- Test
SELECT intern.nimm__zeile( 'tl.befehl', ARRAY['abczk', 'befname'], ARRAY['Ä', 'Ä']
);
SELECT intern.nimm__zeile( 'tl.befehl', ARRAY['befname'], ARRAY[E''] );

-- id automatisch zuordnen
SELECT * FROM tl.nimm__TLWort( 'Änderung', NULL );

-- neuer Eintrag mit id=99
SELECT * FROM tl.nimm__TLWort( 'Hosenbein', '99' );

-- vorhandene id=99 überschreiben
SELECT * FROM tl.nimm__TLWort( 'Hosenbeine', '99' );

-- Test
SELECT * FROM tl.wort;
SELECT * FROM intern.abczk WHERE abczk = 'Änderung';
SELECT * FROM intern.abczk WHERE abczk = 'Hosenbeine';
```

10.15 tl.nimm_erkannte__liste_tlobjekte

Funktionsname
`tl.nimm_erkannte__liste_tlobjekte( IN dateiinhalt varchar, OUT tabname varchar, OUT tabnameid varchar )`

Funktionsparameter/Rückgabewert(e)

`dateiinhalt` Vollständiger Inhalt der einzulesenden TL-Datei

`tabname` Name der Tabelle, in die das erkannte TL-Objekt eingelesen wurde

`tabnameid id` des erkannten TL-Objektes

Funktionsbeschreibung
Ziel der Funktion ist das Zerlegen des Inhaltes `dateiinhalt` einer TL-Datei in TL-Objekte mittels Erweiterter Regulaerer Ausdruecke (ERE). Die erkannten TL-Objekte werden bei Bedarf in die Datenbank eingelesen.

Aus dem übergebenen TL-Dateiinhalt extrahiert die Funktion

- AbcZk welche bei Bedarf in `intern.abczk` (Gueltigkeit siehe `intern.gib__ERE_ABCZK()`),
- RestZk welche bei Bedarf in `intern.restzk` (Gueltigkeit siehe `intern.gib__ERE_RestZK()`),
- TL-Befehle welche bei Bedarf in `tl.befehl` (Gueltigkeit siehe `tl.gib__ERE_TLBefehl()`) und
- TL-Worte welche bei Bedarf in `tl.wort` (Gueltigkeit siehe `tl.gib__ERE_TLWort()`) eingelesen werden.

Die im TL-Wort auftretenden TL-Zweizeichenbefehle müssen im System vorhanden sein.

Sämtliche Zeichenketten, welche nicht diesen Kategorien zuordenbar sind, werden zerlegt und dann entsprechend eingelesen. Zum Beispiel werden TL-Variablennamen, welche ausschliesslich aus Buchstaben bestehen, in `intern.abczk` und nicht in `tl.befehl` eingelesen. Die Ursache liegt in der Unfähigkeit der Funktion, eine Kontextprüfung durchzufuehren.

Mit dem Verzicht des Einlesens von Wortgruppen (siehe `spr.wgruppe`) müssen mehrsprachige Texte mittels TL-Paketen gesetzt werden (z.B. `babel`). Ursache ist auch hier, die Unfähigkeit eine Kontextprüfung durchzuführen.

Beispiele
```
-- Es wird keine Zeile zurückgegeben und nichts eingelesen
SELECT * FROM tl.nimm_erkannte__Liste_TLObjekte( ' );
0
```

```
-- Es wird die Zeichenkette analysiert, die TL-Objekte eingelesen und zurückgegeben
SELECT * FROM tl.nimm_erkannte__Liste_TLObjekte( 'Änderung ÄnderungHosenbein 2006-01-01
#1 Änderung norm-dt_01050_din_en \befehl' );
```

A GNU Free Documentation License

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document freeïn the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of „copyleft“, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The **„Document“**, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as **„you“**. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A **„Modified Version“** of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A **„Secondary Section“** is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **„Invariant Sections“** are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released

under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **„Cover Texts“** are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **„Transparent“** copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not „Transparent“ is called **„Opaque“**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **„Title Page“** means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, „Title Page“ means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section **„Entitled XYZ“** means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **„Acknowledgements“**, **„Dedications“**, **„Endorsements“**, or **„History“**.) To **„Preserve the Title“** of such a section when you modify the Document means that it remains a section „Entitled XYZ“ according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other

conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled „History“, Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled „History“ in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the „History“ section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled „Acknowledgements“ or „Dedications“, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled „Endorsements“. Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled „Endorsements“ or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled „Endorsements“, provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled „History“ in the various original documents, forming one section Entitled „History“; likewise combine any sections Entitled „Acknowledgements“, and any sections Entitled „Dedications“. You must delete all sections Entitled „Endorsements“.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an „aggregate“ if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled „Acknowledgements“, „Dedications“, or „History“, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.
See http://www.gnu.org/copyleft/.
Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License „or any later version“ applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright ©YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled „GNU Free Documentation License“.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the „with...Texts.“ line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software

Verzeichnis der Tabellenspalten und Begriffe

Tabellenspalte / Begriff	Beschreibung	Page List
GNU Free Documentation License	*Der Zweck dieser Lizenz ist es, ein Handbuch, Textbuch oder ein anderes zweckdienliches und nützliches Dokument frei, im Sinne von Freiheit, zu machen; jedermann die Freiheit zu sichern, es zu kopieren und mit oder ohne Änderungen daran, sowohl kommerziell als auch nicht kommerziell weiter zu verbreiten. Weiterhin sichert diese Lizenz einem Autor oder Verleger die Möglichkeit, Anerkennung für seine Arbeit zu erhalten ohne für Änderungen durch Andere verantwortlich gemacht zu werden.* (siehe http://www.giese-online.de/gnufdl-de.html)	3