SAP PRESS

ABAP® Objects

Das umfassende Handbuch

- Konzepte, Sprachelemente und Werkzeuge verständlich erklärt
- Ihr praktisches Nachschlagewerk für alle Programmierfragen
- Inkl. der neuen ABAP-Programmiermodelle für SAP S/4HANA

2., aktualisierte und erweiterte Auflage

Felix Roth

Rheinwerk Publishing

Kapitel 3
Das ABAP Dictionary

Das ABAP Dictionary ist die zentrale Stelle für alle im System vorhandenen Datendefinitionen und damit unerlässlich für jede ABAP-Programmierung. Von Tabellen über Strukturen bis hin zu Suchhilfen – hier werden Sie glücklich!

Das ABAP Dictionary erreichen Sie über den Transaktionscode SE11. Es ist die zentrale Stelle im System, um Datendefinitionen anzulegen und zu verwalten. Es enthält die Beschreibung aller im System vorhandenen Datenstrukturen und stellt diese allen anderen Systemkomponenten auf Bedarf zur Verfügung. Diese Informationen können auch direkt in ABAP-Anweisungen konsumiert werden.

Das ABAP Dictionary ist darüber hinaus die Schnittstelle zur an das SAP-System angebundenen, unter dem System liegenden Datenbank und damit das Tool, um Tabellen bzw. Views auf dieser Datenbank zu erzeugen und zu verwalten. Dazu kann mit ABAP und Open SQL über das ABAP Dictionary auf die Datenbanktabellen zugegriffen werden, ohne diesen Zugriff (z. B. den Aufbau und Abbau der Verbindung) explizit orchestrieren zu müssen. Diese Art des Zugriffs ist einer der Hauptgründe für die Stärke von ABAP, wenn es darum geht, mit großen Datenmengen umzugehen.

Fast der ganze Funktionsumfang des ABAP Dictionarys ist bereits auf dem Einstiegsbildschirm der Transaktion SE11 zu erkennen (siehe Abbildung 3.1), auch wenn sich hinter den Eingabefeldern **View** und **Datentyp** mehr Auswahlmöglichkeiten verstecken.

Sie können mit dem ABAP Dictionary folgende Objekte anzeigen, bearbeiten und anlegen:

- Domänen (siehe Abschnitt 3.1)
- Datenelemente (siehe Abschnitt 3.2)
- Strukturen (siehe Abschnitt 3.3)
- Tabellentypen (siehe Abschnitt 3.4)
- Datenbanktabellen (siehe Abschnitt 3.5)
- Typgruppen (siehe Abschnitt 3.7)
- Views (siehe Abschnitt 3.8)

- Suchhilfen (siehe Abschnitt 3.10)
- das Sperrkonzept (siehe Abschnitt 3.12)

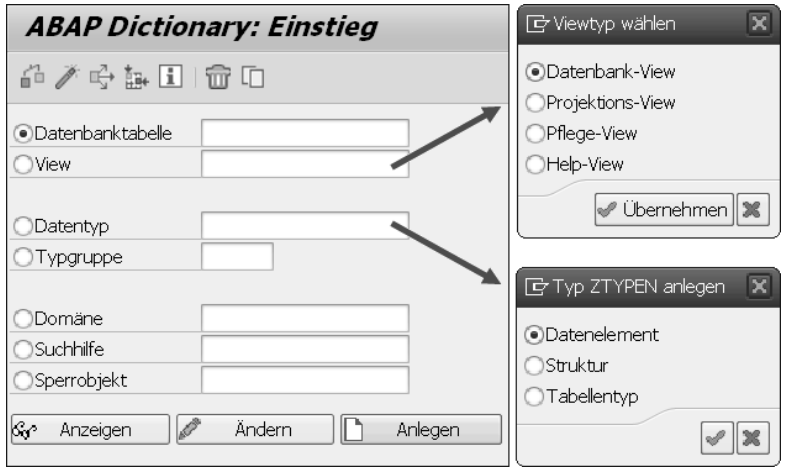

Abbildung 3.1 Einstiegsbildschirm des ABAP Dictionarys

Darüber hinaus bietet das ABAP Dictionary noch etwas versteckte Funktionen an:

- Pflegedialoge (siehe Abschnitt 3.9)
- Datenbank-Utility-Tool (siehe Abschnitt 3.11)
- Indizes (siehe Abschnitt 3.6)

Im Rahmen der vielen Änderungen in der ABAP-Programmierung, die im Zuge der Einführung von SAP HANA vorgenommen wurden, wurde Transaktion SE11 an der einen oder anderen Stelle leicht angepasst. So wurde z. B. die Möglichkeit eingeführt, die Speicherart von Datenbanktabellen zu beeinflussen (siehe Abschnitt 3.5.9) und Volltextindizes zu erstellen (siehe Abschnitt 30.5.1, »Volltextindex anlegen«).

Beispiel

Wie gut ABAP mithilfe des ABAP Dictionarys mit großen Datenmengen umgehen kann, zeigt das folgende Beispiel.

Das folgende ABAP-Programm selektiert 100 Einträge aus der SAP-Standardtabelle für Materialien `MARA`:

```
DATA: lt_mara TYPE TABLE OF mara.
SELECT * FROM mara INTO TABLE lt_mara UP TO 100 ROWS.
```

Das Programm deklariert lediglich eine interne Tabelle auf Basis der im ABAP Dictionary enthaltenen Tabelle `MARA` mit all seinen Feldnamen, Datentypen und Feldlängen. Alle Programme greifen, wenn es um Daten geht, also auf das ABAP Dictionary als zentrale Stelle zu. Das heißt, wenn Sie im ABAP Dictionary Änderungen an einer Tabelle vornehmen, müssen Sie keine Änderungen am Quelltext von Programmen vornehmen. Beim nächsten Aufruf des Programms wird automatisch die Änderung festgestellt, und das Programm wird mit den neuen Informationen neu generiert.

3.1 Domänen

In diesem Abschnitt erläutere ich das dem ABAP Dictionary zugrunde liegende zweistufige Domänenprinzip sowie die Anlage und den Wertebereich einer Domäne.

3.1.1 Das zweistufige Domänenprinzip

Im ABAP Dictionary gibt es ein zweistufiges Domänenprinzip, das technische und semantische Domänen vorsieht:

- Die *technische Domäne* beschreibt den Wertebereich eines Feldes, der durch die Angabe eines eingebauten Datentyps, der Ausgabelänge und eventueller Festwerte festgelegt wird. Im ABAP-Umfeld werden diese technischen Domänen nur *Domänen* genannt.
- Die *semantischen Domänen* auf der anderen Seite weisen den technischen Domänen durch die Vergabe von Texten einen bestimmten Zusammenhang zu. Im ABAP-Umfeld werden diese semantischen Domänen *Datenelement* genannt.

Wie in Abbildung 3.2 dargestellt, kann ein Feld einer Struktur oder Tabelle auf ein Datenelement verweisen, das wiederum auf eine Domäne verweist.

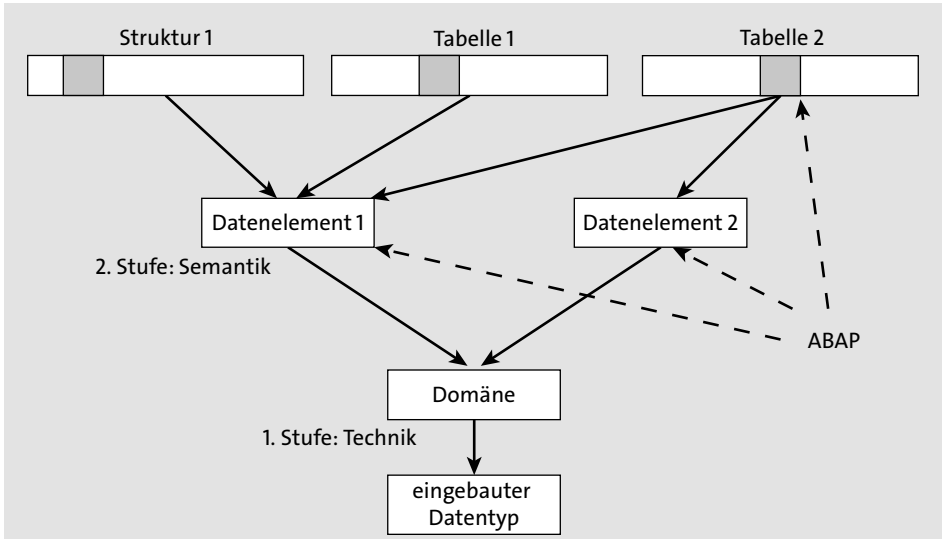

Abbildung 3.2 Das zweistufige Domänenprinzip

Eine Domäne kann demzufolge in mehreren Datenelementen verwendet werden, und ein Datenelement kann in vielen Feldern von Tabellen und Strukturen verwendet werden. Die Domäne fasst also technische Informationen über mehrere Tabellen hinweg zusammen und kann in Form von Datenelementen verschiedene Ausprägungen haben. Darüber hinaus können Sie auf die Datenelemente auch aus ABAP heraus zugreifen und z. B. einen Parameter für eine Eingabemaske definieren.

Beispiel

Eine Identifikationsnummer (ID) wird als Feld nicht nur für eine, sondern in der Regel für viele Tabellen verwendet. Beispielsweise arbeitet eine Universitätsverwaltung mit einer Tabelle für Professoren und einer für Studenten, vielleicht auch einer für Kantinenangestellte. Die Gemeinsamkeit dieser Tabellen: Es wird ein Feld benötigt, das z. B. eine bis zu zehnstellige Identifikations- oder Personalnummer abspeichern kann.

Um diese Aufgabenstellung zu lösen, würden Sie als ABAP-Entwickler nun eine Domäne (CHAR der Länge 10) und für die unterschiedlichen Ausprägungen (Professor, Student) jeweils ein Datenelement anlegen. In ABAP können Sie sich jetzt in mehreren Reports und Selektionsbildschirmen jeweils auf eines der angelegten Datenelemente beziehen und so den dahinterliegenden Text aufrufen und sich damit viel Arbeit sparen. Es dreht sich bei den Domänen und Datenelementen also primär um das Thema der Wiederverwendung.

3.1.2 Domänen anlegen

Um eine Domäne anzulegen, gehen Sie wie folgt vor:

1. Geben Sie den gewünschten Namen für die Domäne auf dem Übersichtsbildschirm der Transaktion SE11 im Feld **Domäne** ein, und klicken Sie anschließend auf **Anlegen**.
2. Es öffnet sich der Domänen-Editor, in dem Sie nun Ihre Domäne ausprägen können (siehe Abbildung 3.3).

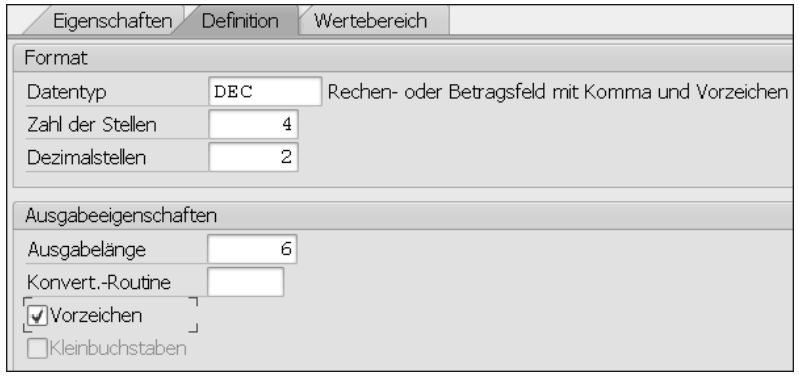

Abbildung 3.3 Definition einer Domäne

Sie müssen mindestens im Eingabefeld **Datentyp** aus den ca. 30 verschiedenen Datentypen einen Datentyp auswählen. Zusätzlich können Sie die **Zahl der Stellen** (im Beispiel mit der Personalnummer waren dies 10) und für numerische Typen die gewünschten **Dezimalstellen** einstellen (bei der Zahl 3,41 sind das z. B. zwei).

3. Unter den **Ausgabeeigenschaften** können Sie zusätzlich die **Ausgabelänge** festlegen. Wichtig ist, dass Sie bei numerischen Datentypen und negativen Zahlen das Häkchen in der Checkbox **Vorzeichen** setzen, da ansonsten das Minuszeichen nicht berücksichtigt wird.
4. Aktivieren Sie die Domäne anschließend über den Button **Aktivieren** () in der Menüleiste.

Darüber hinaus können Sie für Domänen eine Konvertierungsroutine hinterlegen und einen Wertebereich definieren, was in den folgenden Abschnitten beschrieben ist.

3.1.3 Konvertierungsroutinen

Konvertierungsroutinen sind spezielle Routinen, die die Umwandlung von Werten von und zur Datenbank ermöglichen. Im SAP-System gibt es knapp 2.000 Konvertierungsroutinen. Die beiden bekanntesten sind die Konvertierungsroutinen ALPHA, um einem Wert führende Nullen hinzuzufügen oder von ihm zu entfernen, und die Konvertierungsroutine ISOLA, um zweistellige ISO-Sprachschlüssel in einstellige SAP-Sprachschlüssel umzuwandeln und umgekehrt.

Jede Konvertierungsroutine hat einen fünfstelligen Namen und wird als Konvertierungs-Exit in Form von zwei Funktionsbausteinen im System abgelegt. Diese Konvertierungs-Exits finden Sie, indem Sie in Transaktion SE37 (siehe Kapitel 5, »Der Function Builder«) nach Funktionsbausteinen suchen, die wie in Abbildung 3.4 mit dem Namen CONVERSION_EXIT_ anfangen.

Der für Sie interessante Namensbestandteil der Funktionsbausteine in dieser Liste steht zwischen CONVERSION_EXIT_ und _INPUT bzw. _OUTPUT: Dies ist der Name, den Sie im ABAP Dictionary bei der Anlage von Domänen als Konvertierungsroutine im Feld **Konvert.-Routine** angeben müssen (siehe Abbildung 3.3).

Um eigene Konvertierungsroutinen anzulegen, müssen Sie lediglich eine Funktionsbausteingruppe mit zwei Funktionsbausteinen anlegen, einen Funktionsbaustein für die Eingabe und einen für die Ausgabe. Wie Sie eine Funktionsgruppe anlegen können, ist in Abschnitt 5.4 beschrieben.

Die Namen müssen dabei folgendem Schema entsprechen, wobei NAME durch Ihren maximal fünfstelligen Namen ersetzt werden kann:

- CONVERSION_EXIT_NAME_INPUT
- CONVERSION_EXIT_NAME_OUTPUT

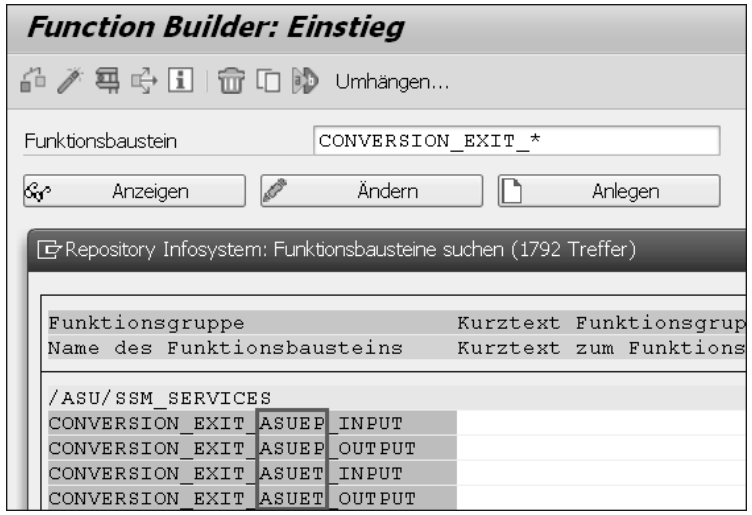

Abbildung 3.4 Konvertierungs-Exits finden

Die beiden Funktionsbausteine müssen folgende Schnittstelle aufweisen:

- einen Eingabeparameter unter IMPORTING mit einem Werteparameter INPUT
- einen Ausgabeparameter unter EXPORTING mit einem Werteparameter OUTPUT

Der einfachste Weg, dies zu erreichen, ist es, einen bestehenden Funktionsbaustein (z. B. CONVERSION_EXIT_ALPHA_INPUT und CONVERSION_EXIT_ALPHA_OUTPUT) zu kopieren und diesen anschließend entsprechend abzuändern.

3.1.4 Wertebereich einer Domäne

Auf der Registerkarte **Wertebereich** zu einer Domäne (siehe Abbildung 3.3) haben Sie die Möglichkeit, die Werte einer Domäne über Festwerte (Einzelwerte), Festwertintervalle oder über eine Wertetabelle einzuschränken. Falls Festwerte für eine Domäne definiert wurden, können diese zusätzlich für eine Eingabeprüfung in Dynpros herangezogen werden. Dies gilt allerdings nur für die Datentypen CHAR und NUMC.

Sie können die Werteüberprüfung für folgende Elemente aktivieren:

- für Parameter mit dem Zusatz VALUE CHECK (siehe Abschnitt 12.2.1, »Parameter«)
- für Selektionsoptionen durch das Event AT SELECTION-SCREEN ON (siehe Abschnitt 12.4, »Ereignisse eines Selektionsbildschirms«)

Sollte keine andere Hilfemöglichkeit, wie z. B. eine Suchhilfe (siehe Abschnitt 3.10), definiert sein, werden die Festwerte auch als Eingabehilfe (F4-Hilfe) angezeigt.

Festwerte und Festwertintervalle

Die Angabe von Festwerten und Intervallen ist nur für Domänen der Datentypen CHAR, NUMC, DEC, INT1, INT2 und INT4 möglich. Beide können beliebig miteinander kombiniert werden.

Wertetabelle

Es können auch alle Werte einer Domäne gegen eine Wertetabelle geprüft werden. Diese Tabelle muss dazu lediglich bei der Angabe des Wertebereichs in der Domäne eingetragen werden. Diese Angabe können Sie auf der Registerkarte **Wertebereich** im Feld **Wertetabelle** machen.

Durch das Eintragen einer Wertetabelle wird aber noch keine Prüfung implementiert. Die Prüfung in Form des Abgleichs mit der Wertetabelle wird erst nach Definition eines Fremdschlüssels wirksam. Eine Fremdschlüsselbeziehung definiert eine Abhängigkeit zwischen zwei Tabellen.

Dazu werden die Schlüsselfelder der ersten Tabelle mit dazu passenden Feldern der zweiten Tabelle verknüpft. Dies wäre beispielsweise möglich, wenn zwei Tabellen jeweils ein Feld MATNR mit einer Materialnummer hätten.

Solange eine solche Verknüpfung nicht definiert wurde, weiß das System nicht, anhand welchen Feldes der angegebenen Wertetabelle die Werte der Domäne geprüft werden soll.

Beispiel

In Abbildung 3.5 ist beispielweise ein Feld CARRID mit dem Datenelement und der gleichnamigen Domäne S_CARR_ID definiert. In der Domäne ist die Wertetabelle SCARR als Prüftabelle eingetragen, die alle möglichen Fluggesellschaften enthält. Durch die für das Feld CARRID über den Button **Fremdschlüssel** () eingetragene Fremdschlüsselbeziehung ist die Prüfung aktiviert worden.

In dem Pop-up-Fenster in Abbildung 3.5 ist erkenntlich, dass die beiden Tabellen über die Felder MANDT und CARRID miteinander verknüpft wurden.

Wenn ein Anwender nun einen Wert für das Feld CARRID auf einer Selektionsmaske eingibt, wird dieser anhand der Wertetabelle SCARR geprüft. Ist der eingegebene Wert nicht in der Prüftabelle enthalten, wird eine Fehlermeldung ausgegeben.

Abbildung 3.5 Wertetabelle aktivieren

3.2 Datenelemente

Um ein Datenelement anzulegen, gehen Sie wie folgt vor:

1. Geben Sie den gewünschten Namen für das Datenelement auf dem Hauptbildschirm der Transaktion SE11 im Feld **Datentyp** ein, und klicken Sie auf **Anlegen**.
2. Wählen Sie in dem Pop-up-Fenster **Datenelement** aus.
3. Es öffnet sich der Datenelement-Editor, in dem Sie nun, wie in Abbildung 3.6 dargestellt, auf der Registerkarte **Datentyp** den Typ spezifizieren können.

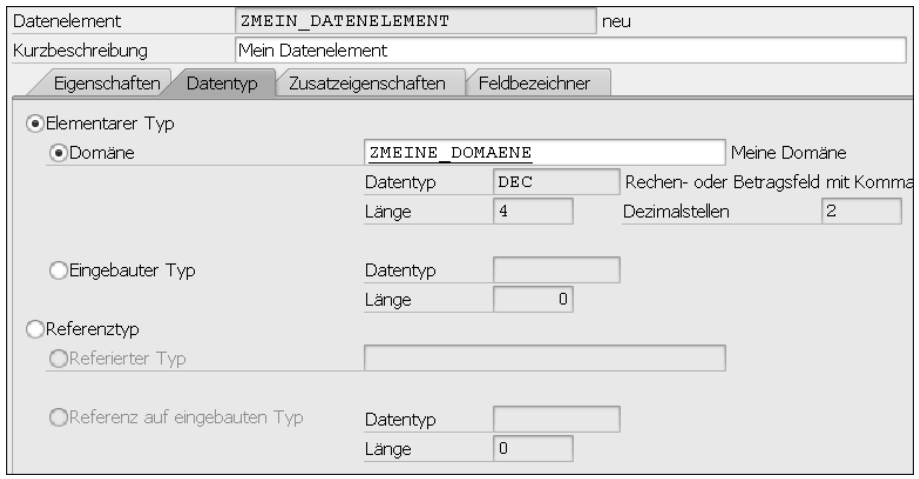

Abbildung 3.6 Datentyp eines Datenelements

Sie können hier aus den folgenden Datentyparten auswählen:

- **Domäne** (siehe Abschnitt 3.1)
- **Eingebauter Typ**
- **Referenztyp** (kann eine Referenz auf eine Klasse, ein Interface oder auf einen eingebauten Typ sein)

4. Danach können Sie Ihr Datenelement bereits aktivieren ( ).

Sie haben darüber hinaus aber noch die Möglichkeit, auf der Registerkarte **Feldbezeichner** die Texte zum Datenelement und auf der Registerkarte **Zusatzeigenschaften** beispielsweise eine Suchhilfe anzugeben. Beide Registerkarten erläutere ich in den folgenden Abschnitten.

3.2.1 Feldbezeichner

Da das Datenelement ja die semantische Bedeutung zur Typdefinition liefert (siehe Abschnitt 3.1.1, »Das zweistufige Domänenprinzip«), können Sie dem Datenelement auf der Registerkarte **Feldbezeichner** verschiedene Texte zuordnen (siehe Abbildung 3.7).

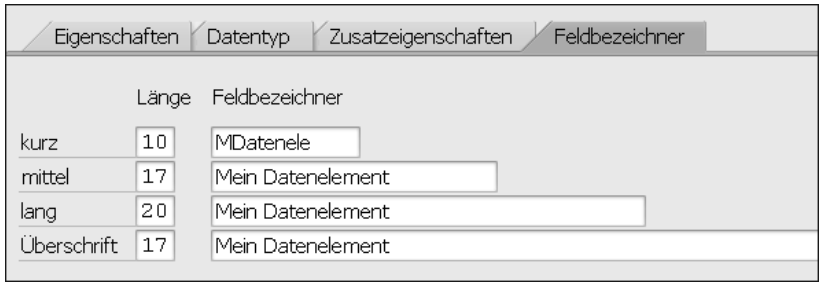

Abbildung 3.7 Feldbezeichner eines Datenelements

Um die Texte zu pflegen, geben Sie sie in die Eingabefelder **Feldbezeichner** ein, und bestätigen Sie Ihre Eingabe anschließend mit ↵. Die Länge der Eingabefelder gibt dabei die maximale Länge der Texte vor.

3.2.2 Übersetzung

Fast alle in Transaktion SE11 zu pflegenden Objekte haben Texte (z. B. eine Kurzbeschreibung), die sich übersetzen lassen. Doch nirgends hat dies eine wichtigere Bedeutung als bei den Feldbezeichnern eines Datenelements.

Diese Texte werden später beispielsweise beim Aufbau einer ALV-Tabelle als Spaltentexte verwendet (siehe Kapitel 19, »Tabellenanzeige mit dem SAP List Viewer (ALV)«) oder auf einem Selektionsbildschirm in Form eines Parameters oder eines Select-Op-

tion-Namens dargestellt (siehe Kapitel 12, »Reports und Selektionsbildschirme«). Ist der Text übersetzt, wird abhängig von der Anmeldesprache des Nutzers jeweils die passende Übersetzung angezeigt.

Um einen Text zu einem Datenelement zu übersetzen, gehen Sie wie folgt vor:

1. Rufen Sie das Datenelement im ABAP Dictionary auf, und wählen Sie **Springen • Übersetzung** im Hauptmenü.
2. Wählen Sie, wie in Abbildung 3.8 dargestellt, im sich öffnenden Pop-up-Fenster die gewünschte Zielsprache aus.

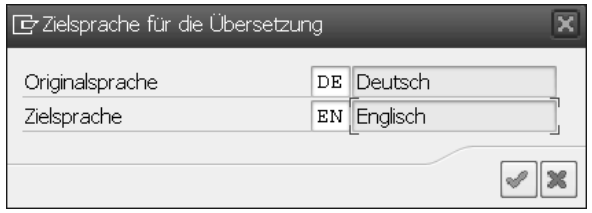

Abbildung 3.8 Angabe der Zielsprache

3. In der nächsten Maske werden Ihnen nun alle zu übersetzenden Texte angezeigt. Diese können Sie jeweils unter dem originalen Textblock übersetzen, indem Sie den übersetzten Text eintragen, wie in Abbildung 3.9 dargestellt.

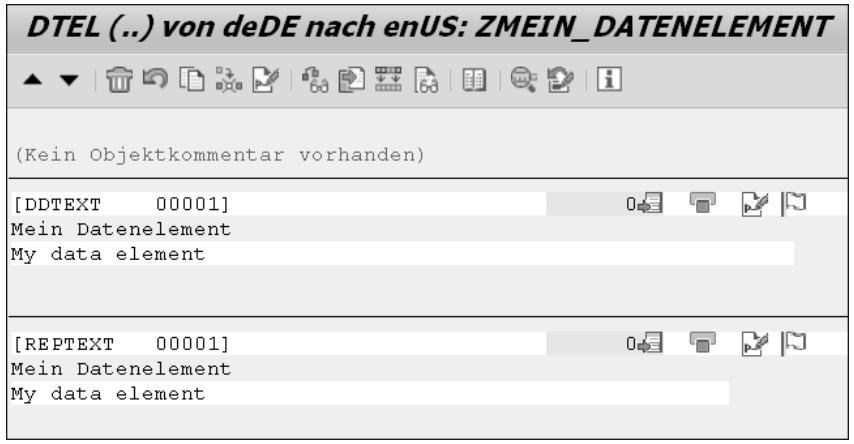

Abbildung 3.9 Übersetzung von Textelementen

4. Sie haben darüber hinaus die Möglichkeit, sich über den Button **Quelltexte in SAPterm suchen** (📖) auf der Funktionsleiste Vorschlagswerte aus der SAPterm-Datenbank anzeigen zu lassen.
5. Wenn Sie anschließend auf **Speichern** (💾) klicken, werden Ihnen Ihre eingegebenen Texte gelb angezeigt.

Prinzipiell können Texte folgende Status haben:

- Rot: neuer Text
- Gelb: beanstandeter Text
- Grün: korrekt übersetzter Text

Wenn Sie lediglich eine Übersetzung benötigen, reicht der Status Gelb aus. Damit verzichten Sie allerdings auf die Wiederverwendung von bereits angefertigten Übersetzungen, da damit Ihre Übersetzung nicht in den Vorschlagspool übernommen wird.

Möchten Sie Ihre Übersetzung dagegen in den Vorschlagspool übernehmen, um z. B. die Wiederverwendung zu ermöglichen, können Sie auf den Button **Vorschlag direkt anlegen** rechts neben dem gelben Statusfeld klicken. Damit wechselt der Status von Gelb auf Grün. Speichern Sie gegebenenfalls erneut.

3.2.3 Zusatzeigenschaften

Auf der Registerkarte **Zusatzeigenschaften** haben Sie die Möglichkeit, dem Datenelement eine Suchhilfe zuzuordnen (siehe Abschnitt 3.10). Über eine Parameter-ID kann Ihr Feld mit einem Vorschlagswert aus dem SAP-Memory gefüllt werden. Für jeden Benutzer können Sie einen solchen Wert in dessen Benutzerstammdaten auf der Registerkarte **Parameter** pflegen. Die Benutzerstammdaten erreichen Sie über den Pfad **System • Benutzervorgaben** im Hauptmenü.

3.3 Strukturen

Eine Struktur ist ein Verbund einzelner Felder. Wie Sie eine Struktur programmatisch definieren, ist in Abschnitt 7.4.3 erläutert. Um eine Struktur im ABAP Dictionary anzulegen, gehen Sie wie folgt vor:

1. Geben Sie den gewünschten Namen auf dem Hauptübersichtsbildschirm der Transaktion SE11 im Feld **Datentyp** ein. Klicken Sie anschließend auf **Anlegen**.
2. Wählen Sie im sich öffnenden Pop-up-Fenster den Datentyp **Struktur** aus.
3. Es öffnet sich der Struktur-Editor, in dem Sie nun, wie in Abbildung 3.10 dargestellt, auf der Registerkarte **Komponenten** die gewünschten Felder Ihrer Struktur eintragen können. Für jedes Feld müssen Sie die **Typisierungsart** und einen Typ in Form eines Datenelements oder anderer Strukturen (*tiefe Strukturen*) vergeben.
4. Danach können Sie Ihre Struktur bereits **Aktivieren** (✎).

Die in Abbildung 3.10 angegebene Struktur wird allerdings eine Warnung und eine Fehlermeldung produzieren:

- **Warnung**
  Die Erweiterungskategorie fehlt.
- **Fehler**
  Für das Feld `BRGEW` fehlen die Referenztabelle und das Referenzfeld.

Dies schauen wir uns in den folgenden beiden Abschnitten an.

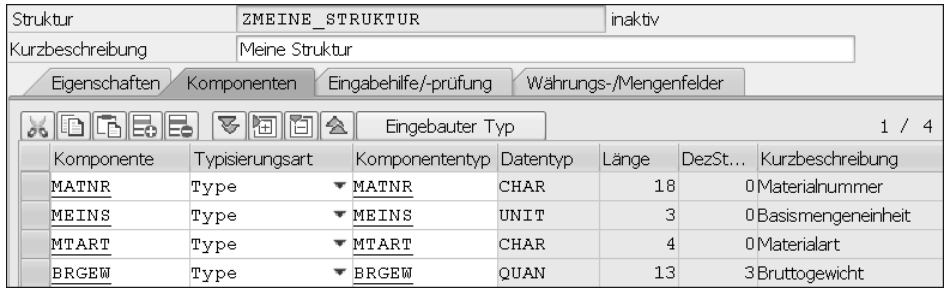

Abbildung 3.10 Komponenten einer Struktur

3.3.1 Erweiterungskategorie

Die Erweiterungskategorie können Sie über den Pfad **Zusätze • Erweiterungskategorie definieren** im Hauptmenü definieren (siehe Abbildung 3.11). Damit bestimmen Sie, ob und wie die Struktur erweitert werden darf.

Die folgenden Optionen stehen Ihnen dazu zur Verfügung:

- **beliebig erweiterbar**
  Die Struktur darf um beliebige Komponenten erweitert werden.
- **erweiterbar und zeichenartig oder numerisch**
  Die Struktur darf erweitert werden, die Erweiterung darf aber keine tiefen Datentypen enthalten.
- **erweiterbar und zeichenartig**
  Die Struktur darf mit zeichenartigen Komponenten (`c`, `n`, `d` oder `t`) erweitert werden.
- **nicht erweiterbar**
  Die Struktur darf nicht erweitert werden.
- **nicht klassifiziert**
  Undefiniert, kann für einen Übergangszustand verwendet werden.

Darf eine Struktur erweitert werden, kann dies mithilfe einer Append-Struktur geschehen. Mehr Informationen zu Append-Strukturen finden Sie in Abschnitt 21.4.7, »Strukturerweiterungen«.

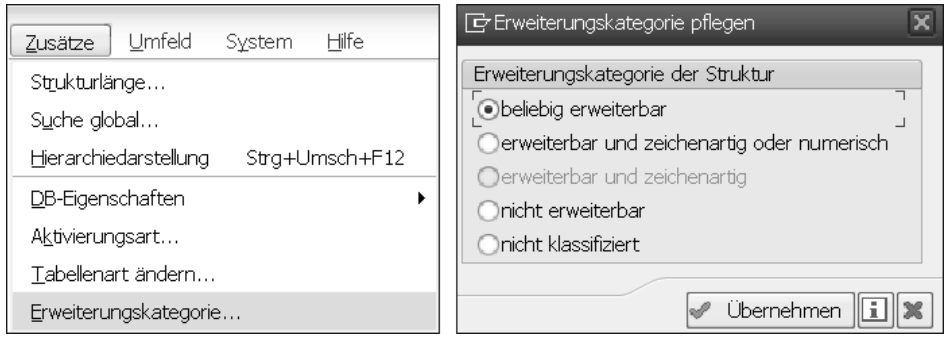

Abbildung 3.11 Erweiterungskategorie einer Struktur

3.3.2 Referenztabellen und das Referenzfeld

Mengen- und Währungsfelder benötigen immer eine Zuweisung zu einer Mengen- bzw. Währungseinheit, damit das System weiß, in welcher Form die Menge bzw. die Währung darzustellen ist. Diese Zuweisung können Sie auf der Registerkarte **Währungs-/Mengenfelder** vornehmen (siehe Abbildung 3.12). Der Verweis kann dabei auf jede beliebig andere Tabelle bzw. Struktur zeigen.

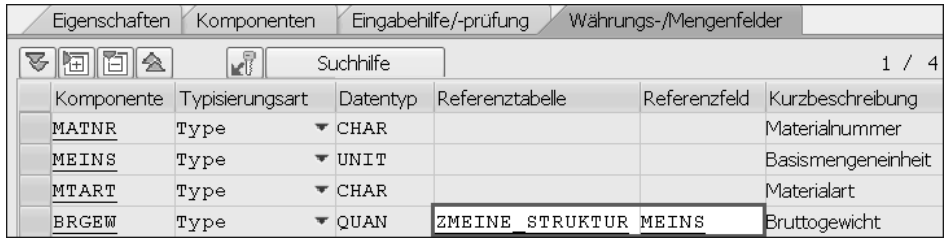

Abbildung 3.12 Verweis auf Einheiten

3.4 Tabellentypen

Tabellentypen sind spezielle Typen, die den Aufbau einer internen Tabelle beschreiben. Das Besondere ist, dass eine interne Tabelle auf Basis des Tabellentyps einfach mit dem Zusatz `TYPE` definiert werden kann und dass dazu nicht der Zusatz `TYPE TABLE OF` benötigt wird. Dies ist insbesondere in Schnittstellen von z. B. Funktionsbausteinen und Methoden von Klassen wichtig, da dort nicht `TYPE TABLE OF`, sondern nur `TYPE` angegeben werden kann.

Um einen Tabellentyp anzulegen, gehen Sie wie folgt vor:

1. Geben Sie den gewünschten Namen des Tabellentyps auf dem Hauptübersichtsbildschirm der Transaktion SE11 im Feld **Datentyp** ein. Klicken Sie anschließend auf **Anlegen**.

2. Wählen Sie im Pop-up-Fenster den Datentyp **Tabellentyp** aus.
3. Es öffnet sich der Tabellentyp-Editor, in dem Sie nun, wie in Abbildung 3.13 dargestellt, auf der Registerkarte **Zeilentyp** den gewünschten Aufbau Ihres Tabellentyps anhand einer Struktur definieren können.

Abbildung 3.13 Zeilentyp eines Tabellentyps

4. Auf der Registerkarte **Initialisierung und Zugriff** können Sie die Art der internen Tabelle für den Tabellentyp auswählen (siehe Abschnitt 8.1, »Tabellenarten«).
5. Auf den Registerkarten **Primärschlüssel** und **Sekundärschlüssel** können Sie nun noch die Schlüsselfelder definieren.
6. Anschließend können Sie Ihren Tabellentyp **Aktivieren** (🖉) und z. B. in Methodensignaturen oder Funktionsbausteinschnittstellen verwenden.

Neben den hier beschriebenen globalen Tabellentypen im ABAP Dictionary können Sie Tabellentypen auch programmatisch mit ABAP für einen lokalen Einsatz definieren. Mehr Informationen dazu finden Sie in Abschnitt 8.2, »Interne Tabellen definieren«.

3.4.1 Ranges-Tabellentypen anlegen

Neben den Tabellentypen für Standardtabellen können Sie im ABAP Dictionary auch globale Ranges-Tabellen als Typ definieren (siehe Abschnitt 8.2.3):

1. Legen Sie dazu, wie in Abschnitt 3.4, »Tabellentypen«, beschrieben, einen neuen Tabellentyp an, und wählen Sie **Bearbeiten • Als Ranges-Tabellentyp definieren** im Hauptmenü (siehe Abbildung 3.14).

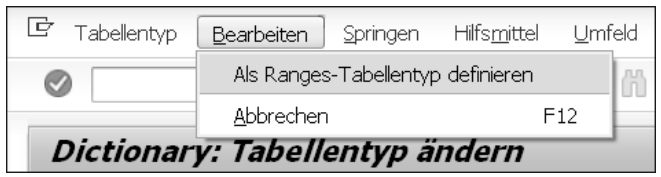

Abbildung 3.14 Einen Ranges-Tabellentyp anlegen

2. Tragen Sie im folgenden Bildschirm, wie in Abbildung 3.15 dargestellt, eine **Kurzbeschreibung** und ein **Datenelement** bzw. einen eingebauten Typ ein, für das oder den Sie den Ranges-Tabellentyp definieren wollen.
3. Vergeben Sie anschließend noch einen Namen für die dahinterliegende Struktur im Feld **Strukturierter Zeilentyp**.
4. Speichern (💾) Sie nun Ihren Ranges-Tabellentyp, da sonst die Anlage des strukturierten Zeilentyps nicht funktioniert.

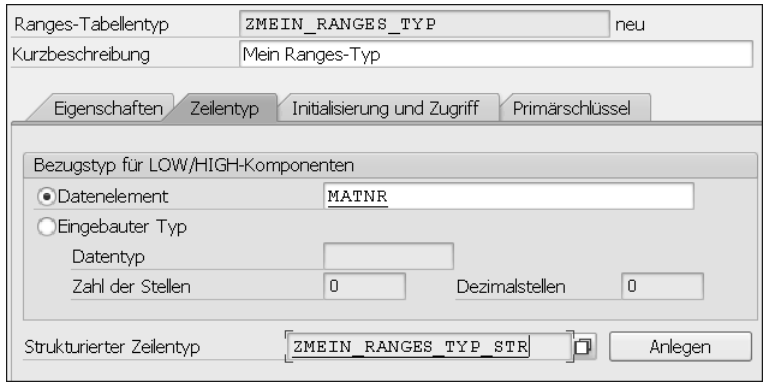

Abbildung 3.15 Anlage des strukturierten Zeilentyps

5. Nun können Sie auf den Button **Anlegen** klicken, wodurch eine neue Struktur auf Basis Ihrer Eingabe angelegt wird. Wie Sie in Abbildung 3.16 sehen können, ist dies die Standardstruktur einer Ranges-Tabelle.

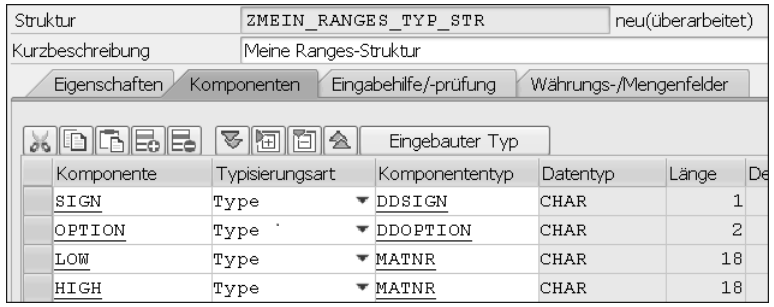

Abbildung 3.16 Komponenten der Ranges-Struktur

6. Vergeben Sie hier eine **Kurzbeschreibung**, aktivieren (🖉) Sie die Struktur, und wechseln Sie über die Navigationsleiste (↩) zurück zum Bildschirm **Tabellentyp ändern.**
7. Aktivieren (🖉) Sie auch dort Ihren Ranges-Tabellentyp.

3.5 Datenbanktabellen

Eine der Hauptfunktionen des ABAP Dictionarys ist die Verwaltung der zentralen Datenbanktabellen des SAP-Systems. Eine Datenbanktabelle beinhaltet eine Menge von Daten, die wie in einem Excel-Sheet in Zeilen und Spalten strukturiert sind. Die Spaltennamen definieren, was für Daten in jeder Spalte stehen sollen, während die einzelnen Zeilen die einzelnen Datensätze repräsentieren. Über das ABAP Dictionary als Schnittstelle zur an das SAP-System angebundenen Datenbank können Sie sowohl die Strukturen als auch die Daten aller Datenbanktabellen anzeigen lassen. Zusätzlich können Sie auch eigene Datenbanktabellen für Ihre Entwicklungen anlegen und so ermöglichen, dass Daten langfristig gespeichert werden.

3.5.1 Datenbanktabellen anzeigen

Um eine Datenbanktabelle anzuzeigen, gehen Sie wie folgt vor:

1. Tragen Sie den Namen der gewünschten Tabelle auf dem Hauptübersichtsbildschirm der Transaktion SE11 im Feld **Datenbanktabelle** ein.
2. Anschließend gelangen Sie durch einen Klick auf den Button **Anzeigen** zur Definition der Tabelle. Wenn Sie nun den Inhalt der Datenbanktabelle sehen möchten, klicken Sie in der Funktionsleiste auf den Button **Datenanzeige** ().
3. In der folgenden Selektionsmaske können Sie die anzuzeigende Datenmenge einschränken. Nehmen Sie hier keine Einstellung vor, werden standardmäßig 200 Datensätze angezeigt.
4. Nun können Sie innerhalb der Datenbanktabelle durch Eingabe von Suchkriterien nach Datensätzen suchen. Klicken Sie auf den Button **Anzahl Einträge**, wird Ihnen angezeigt, wie viele Einträge zu Ihren Suchkriterien passen. Klicken Sie auf **Ausführen** (), um Ihre Suche innerhalb der Datenbanktabelle zu starten.

3.5.2 Datenbanktabellen anlegen

Zum Anlegen einer neuen Datenbanktabelle tragen Sie den gewünschten Namen der Tabelle ebenfalls auf dem Hauptübersichtsbildschirm der Transaktion SE11 im Feld **Datenbanktabelle** ein, und klicken Sie anschließend auf **Anlegen**. Für den Namen gibt es keine speziellen Namenskonventionen, er muss lediglich im Kundennamensraum (z. B. `Z` oder `Y`) liegen. Es öffnet sich nun der Tabellen-Editor (siehe Abbildung 3.17).

Zur Anlage einer Tabelle müssen Sie die folgenden Einstellungen vornehmen, auf die ich in den folgenden Abschnitten näher eingehe:

- Auf der Registerkarte **Auslieferung und Pflege:**
  - **Auslieferungsklasse:** Soll die Datenbanktabelle einen Transportanschluss haben (siehe Abschnitt 3.5.3)?

  - **Data Browser/Tabellensicht-Pflege:** Wie soll auf die Datenbanktabelle zugegriffen werden können (siehe Abschnitt 3.5.4)?
- In den technischen Einstellungen:
  - **Datenart:** Was soll in der Tabelle gespeichert werden (siehe Abschnitt 3.5.5)?
  - **Größenkategorie:** Wie viele Daten werden erwartet (siehe Abschnitt 3.5.6)?
  - **Pufferung** (optional): Darf die Datenbanktabelle gepuffert werden (siehe Abschnitt 3.5.7)?
- Ausprägen der Tabellenfelder auf der Registerkarte **Felder**: Welche Spalten soll die Datenbanktabelle haben (siehe Abschnitt 3.5.8)?

Abbildung 3.17 Registerkarte »Auslieferung und Pflege« im Tabellen-Editor

Darüber hinaus gibt es seit der Einführung von SAP HANA eine neue Registerkarte **Speicherart**, auf die ich in Abschnitt 3.5.9 eingehe.

3.5.3 Auslieferungsklasse

Eine Auslieferungsklasse wählen Sie auf der Registerkarte **Auslieferung und Pflege** (siehe Abbildung 3.17). Die Auslieferungsklasse bestimmt, wie sich die Datenbanktabelle bei der Installation, beim Upgrade, bei einer Mandantenkopie und beim Transport zwischen Kundensystemen verhält. Die Frage, die Sie sich hier also stellen müssen, lautet: Wie soll sich die Datenbanktabelle beim Transport verhalten? Möchten Sie die Datenbanktabelle in jedem System (z. B. Entwicklungs-, Qualitätssicherungs- und Produktivsystem) einzeln pflegen, oder soll die Datenbanktabelle mithilfe der Transporttechnologie über alle Systeme hinweg den gleichen Inhalt haben?

Wenn Sie eine Tabelle mit der Auslieferungsklasse »A« anlegen (siehe Abbildung 3.18), kann die Datenbanktabelle in jedem System über einen *Pflegedialog* geändert werden. Legen Sie Ihre Datenbanktabelle dagegen mit der Auslieferungsklasse »C« an, kann diese Datenbanktabelle nur in Entwicklungssystemen geändert werden (wenn das System auf **änderbar** gesetzt ist), und die Inhalte müssen über das *Transportwesen* im Rahmen eines Customizing-Auftrags transportiert werden.

Wie Sie einen Pflegedialog anlegen, ist in Abschnitt 3.9, »Pflegedialoge«, beschrieben, während Sie in Kapitel 17, »Das Transportwesen«, eine Erläuterung des Transportmechanismus finden.

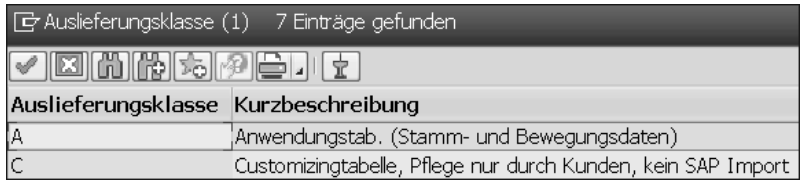

Abbildung 3.18 Auslieferungsklasse einer Tabelle

3.5.4 Tabellensicht-Pflege

Auf der Registerkarte **Auslieferung und Pflege** (siehe Abbildung 3.17) stellen Sie die Berechtigungen für die Anzeige bzw. Pflege Ihrer Datenbanktabelle ein. Die grundlegende Frage, die Sie sich hier stellen müssen, lautet: Wie soll auf die Datenbanktabelle zugegriffen werden können?

Hier haben Sie die folgenden Optionen zur Auswahl:

- **Anzeige/Pflege eingeschränkt erlaubt**
  Die Anzeige der Datenbanktabelle ist nur im ABAP Dictionary (Transaktion SE11) und in der allgemeinen Tabellenanzeige (Transaktion SE16 bzw. SE16N) möglich.
- **Anzeige/Pflege erlaubt**
  Die Tabelle kann in der allgemeinen Tabellenanzeige angezeigt und in der Tabellensicht-Pflege (Transaktion SM30) über einen Pflegedialog (siehe Abschnitt 3.9, »Pflegedialoge«) gepflegt werden.
- **Anzeige/Pflege nicht erlaubt**
  Anzeige bzw. Pflege ist nur über ABAP-Anweisungen (d. h. via Open SQL) möglich.

3.5.5 Datenart

Die Art der in Ihrer Datenbanktabelle gespeicherten Daten können Sie in den technischen Einstellungen auswählen (siehe Abbildung 3.19). Klicken Sie dazu auf den Button **Technische Einstellungen** in der Funktionsleiste.

Die wichtigsten Datenarten sind:

- **APPL0** für Stammdaten
- **APPL1** für Bewegungsdaten
- **APPL3** für Organisations- und Customizing-Daten

Abbildung 3.19 Technische Einstellungen einer Tabelle

Die Frage, die Sie sich hierbei stellen müssen, lautet: Welche Daten wollen Sie speichern? Sind es Daten, die sich häufig ändern (*Bewegungsdaten*), sind es Daten, auf die zwar häufig lesend zugegriffen wird, die sich jedoch sehr selten ändern (*Stammdaten*), oder sind es Daten, die für das Customizing des Systems benötigt werden? Die Auswahl hat für Sie als Programmierer keine großen Auswirkungen, Sie beeinflusst lediglich, wo die Daten auf der Datenbank abgelegt werden, und dies auch nur für die Datenbanksysteme Oracle und Informix.

3.5.6 Größenkategorie

Die Größenkategorie Ihrer Datenbanktabelle können Sie ebenfalls in den technischen Einstellungen auswählen (siehe Abbildung 3.19). Hier geht es darum, anzugeben, wie viel Speicherplatz Ihre Tabelle in Zukunft vermutlich einnehmen wird.

Beim Anlegen der Datenbanktabelle wird aufgrund der hier angegebenen Größe ein initialer Speicherplatz auf der Datenbank reserviert. Hintergrund ist, dass aufwendige Reorganisationen des Speicherplatzes vermieden werden sollen, die entstehen, wenn der reservierte Speicherplatz überschritten wurde. Die Frage, die Sie sich in diesem Kontext stellen müssen, lautet also: Wie viele Daten werden vermutlich in der Datenbanktabelle gespeichert werden?

3.5.7 Pufferung

Die Pufferung können Sie ebenfalls in den technischen Einstellungen konfigurieren. Der *Puffer* ist ein spezieller Bereich auf dem Applikationsserver, in dem Datensätze Ihrer Tabelle vorgehalten werden. Greift eine SQL-Anweisung auf eine gepufferte Tabelle zu, wird geprüft, ob die angefragten Daten sich in diesem Pufferbereich befinden. Ist dies der Fall, werden die Daten direkt aus dem Puffer gelesen. Ist dies nicht der Fall, werden die Daten von der Datenbank gelesen und dabei in den Puffer geladen.

Die Pufferung einer Tabelle erhöht die Performance bei jedem Zugriff auf die in der Tabelle enthaltenen Datensätze, da nicht jedes Mal auf die Datenbank zugegriffen werden muss.

Ob Sie die Pufferung für eine Datenbanktabelle zulassen sollten oder nicht, hängt davon ab, wie später mit der Tabelle gearbeitet wird. Sind viele lesende Zugriffe abzusehen, lohnt sich eine Pufferung, bei vielen schreibenden Zugriffen lohnt sich dagegen keine Pufferung. Wenn Sie die Pufferung einschalten, müssen Sie auch eine Pufferungsart auswählen. Die Fragen, die Sie sich an dieser Stelle stellen müssen, lauten also: Erfolgen auf meine Datenbanktabelle überwiegend lesende Zugriffe, und möchte ich die Geschwindigkeit des Lesezugriffs erhöhen? Zwei grundlegende Pufferungsarten stehen Ihnen zur Verfügung (siehe Abbildung 3.20):

- **Einzelsätze gepuffert**
  Lohnt sich bei großen Tabellen mit vielen einzelnen Zugriffen, da auch für nicht vorhandene Einträge im Puffer ein Vermerk angelegt wird.
- **vollständig gepuffert**
  Je kleiner eine Tabelle ist, je häufiger sie gelesen und je seltener in sie geschrieben wird, desto günstiger ist es, sie vollständig zu puffern

Abbildung 3.20 Pufferung einer Tabelle

3.5.8 Felder ausprägen

Nachdem Sie die Eigenschaften und technischen Einstellungen gepflegt haben, können Sie sich endlich dem wichtigsten Punkt bei der Anlage einer Datenbanktabelle widmen: den Tabellenfeldern auf der Registerkarte **Felder**. Wie in Abbildung 3.21 zu sehen, können Sie hier wie bei der Anlage von Strukturen (siehe Abschnitt 3.3) Ihre gewünschten Felder eintragen und über Datenelemente typisieren.

Wenn Ihre Tabelle mandantenabhängig sein soll (das ist in der Regel der Fall), müssen Sie als erstes Feld das Feld `MANDT` vom Typ `MANDT` hinzufügen. Dieses Feld wird dann automatisch bei jeder Open-SQL-Anweisung mit dem aktuellen Mandanten gefüllt bzw. beim lesenden Zugriff entsprechend verarbeitet.

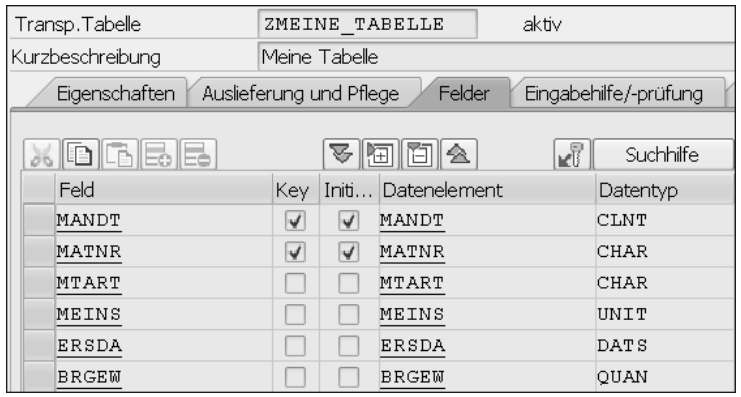

Abbildung 3.21 Felder einer Tabelle ausprägen

Summe aller Feldlängen

Eine Tabelle darf maximal 749 Felder enthalten, wobei die Summe aller Feldlängen auf 4.030 begrenzt ist. Die Summe der Feldlängen ergibt sich aus der Anzahl von Bytes der nicht zeichenartigen Felder und der Anzahl der Zeichen der zeichenartigen Felder. Die Summe der Feldlängen wird mit ABAP 7.53 nicht mehr geprüft, sondern nur noch die Anzahl der Spalten. Diese wurden bei der Speicherart **Row Store** auf 1.000 und bei **Column Store** (sowie Datenbank-Views und CDS Views) auf 1.500 Spalten erhöht.

3.5.9 Speicherart von Datenbanken mit SAP HANA

Mit SAP NetWeaver Application Server ABAP 7.40 wurde im ABAP Dictionary eine neue Registerkarte **DB spezifische Eigenschaften** eingeführt (siehe Abbildung 3.22). Hier können Sie steuern, mit welcher Speicherart eine Datenbanktabelle auf dem Datenbanksystem angelegt werden soll.

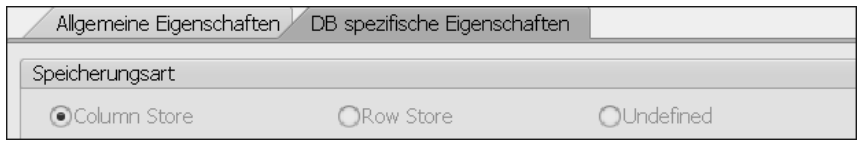

Abbildung 3.22 Speicherart einer Tabelle

Sie können zwischen drei verschiedenen Speicherarten wählen:

- **Column Store:** spaltenbasierte Speicherung als die neue Speicherart der SAP-HANA-Datenbank (siehe Kapitel 30, »SAP HANA«)
- **Row Store:** zeilenbasierte Speicherung als herkömmlicher Standard
- **Undefined:** andere

Diese Einstellung kann jederzeit nach der Erstellung der Tabelle geändert werden. Die Tabelle wird anschließend entsprechend umgesetzt.

3.6 Indizes

Mit Indizes können Sie das Durchsuchen einer Tabelle nach Datensätzen beschleunigen. Ein Index kann als sortierte Teilmenge einer Datenbanktabelle verstanden werden und ermöglicht so einen schnelleren Zugriff auf die Datensätze. Im SAP-System wird zwischen folgenden Arten von Indizes unterschieden:

- **Primärindizes**
  Der Primärindex besteht aus dem Primärschlüssel einer Tabelle sowie einem Zeiger auf ihre Nicht-Schlüsselfelder, damit diese bei Bedarf schnell nachgelesen werden können. Der Primärindex wird beim Anlegen der Tabelle auf der Datenbank automatisch erstellt.
- **Sekundärindizes**
  Darüber hinaus können Sie im ABAP Dictionary sogenannte Sekundärindizes anlegen. Diese Indizes sind notwendig, wenn häufig über Nicht-Schlüsselfelder auf die Tabelle zugegriffen wird, da hier der Primärindex nicht genutzt werden kann, der ja nur Schlüsselfelder enthält.

  Es können mehrere dieser Sekundärindizes für eine Tabelle existieren. Erst zur Laufzeit wird vom Datenbanksystem entschieden, welcher Index verwendet werden muss.

Sekundärindizes und SAP HANA

Durch die Einführung von SAP HANA und die spaltenorientierte Speicherung haben Sekundärindizes an Bedeutung verloren. Weitere Informationen dazu finden Sie unter Regel 4 im Abschnitt »Die goldenen Regeln für SAP HANA« in Abschnitt 9.1.

Indizes anlegen

Um einen Index anzulegen, gehen Sie wie folgt vor:

1. Lassen Sie sich die Definition der Tabelle, für die Sie einen Index anlegen möchten, in Transaktion SE11 anzeigen (siehe Abschnitt 3.5.1, »Datenbanktabellen anzeigen«).
2. Klicken Sie im Tabellen-Editor in der Funktionsleiste auf **Indizes**.
3. In der Übersicht der vorhandenen Indizes für die Tabelle klicken Sie nun, wie in Abbildung 3.23 dargestellt, in der Menüleiste auf den Button **Anlegen** ( ) und wählen aus dem Auswahlmenü die Aktion **Index anlegen** aus.

Abbildung 3.23 Index zu einer Tabelle anlegen

4. Hier können Sie nun angeben, für welches Datenbanksystem der Index angelegt werden soll und welche Felder der Index beinhalten soll (siehe Abbildung 3.24). Wenn Ihr Index den Primärschlüssel enthält, also eine Zeile eindeutig identifiziert, können Sie auch die Option **Unique-Index** auswählen.

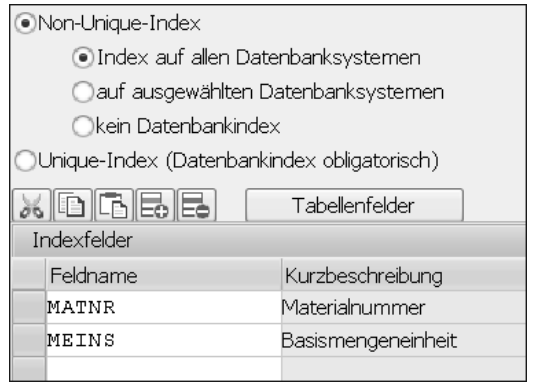

Abbildung 3.24 Definition der Indexfelder

5. Aktivieren Sie Ihren Index anschließend über den Button **Aktivieren** ( ).

3.7 Typgruppen

Eine Typgruppe ist ein historisch bedingtes Konstrukt und dient der Sammlung mehrerer Typen. Es war nötig, da es vor Release 4.5A im ABAP Dictionary keine eigenständigen Datentypen gab, auf die in ABAP-Programmen mit dem `TYPE`-Zusatz verwiesen werden konnte.

Um eine Typgruppe anzulegen, geben Sie den gewünschten Namen auf dem Hauptübersichtsbildschirm der Transaktion SE11 im Feld **Typgruppe** ein und klicken anschließend auf **Anlegen**. Im sich öffnenden Pop-up-Fenster tragen Sie eine **Kurzbeschreibung** ein und speichern die Typgruppe mit einem Klick auf **Sichern**. Innerhalb der angelegten Typgruppe können Sie nun über den Editor Ihre benötigten Typen jeglicher Art definieren (siehe Abbildung 3.25).

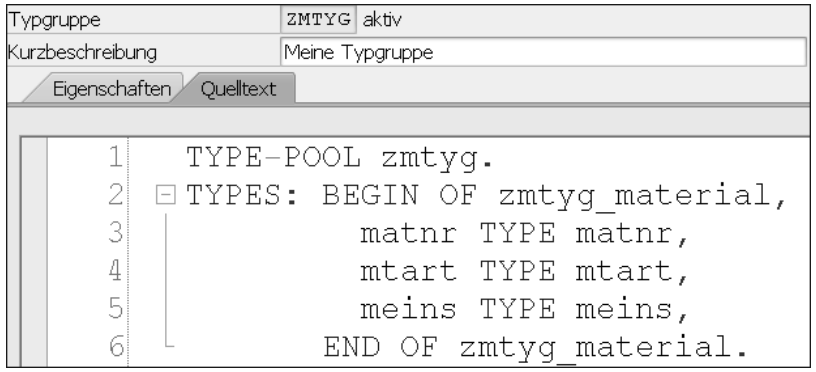

Abbildung 3.25 Eine Typgruppe und seine Typen

Im ABAP-Programm können Sie Ihre Typgruppe im Beispiel aus Abbildung 3.25 mit der folgenden Anweisung einbinden:

```
TYPE-POOLS zmtyg.
```

3.8 Views

Ein View ist eine Zusammenfassung ausgewählter Felder aus mehreren Tabellen, ähnlich einem Join zur Verknüpfung von Datenbanktabellen (siehe Abschnitt 9.2.12, »JOIN: Verknüpfung«). Der Vorteil von Views gegenüber selbst mit ABAP programmierten Joins ist, dass sie rein über Definitionsmasken und nicht durch Programmierung definiert werden und dass Views durch die zentrale Anlage im ABAP Dictionary systemweit wiederverwendet werden können. Die so ausgewählte Schnittmenge wird als eigene Struktur definiert und kann in ABAP-Programmen mit Open SQL konsumiert werden.

Die Daten eines Views werden wie bei einem Join nicht physisch in einer separaten Tabelle gespeichert, sondern als Teilmenge zur Laufzeit aus einer oder mehreren Tabellen durch Selektion (Weglassen von Spalten) und Projektion (Weglassen von Zeilen) abgeleitet.

Um einen View anzulegen, geben Sie den gewünschten Namen auf dem Hauptübersichtsbildschirm der Transaktion SE11 im Feld **View** ein und klicken anschließend auf **Anlegen**. Im sich öffnenden Pop-up-Fenster haben Sie nun folgende View-Typen zur Auswahl:

- Datenbank-View (siehe Abschnitt 3.8.1): normale Umsetzung einer Tabellenverknüpfung
- Projektions-View (siehe Abschnitt 3.8.2): Hier ist keine Verknüpfung von Tabellen möglich; dieser View-Typ dient dem Ausblenden von Spalten einer Tabelle.

- Pflege-View (siehe Abschnitt 3.8.3): Pflege-Views ermöglichen die Pflege von über mehrere Tabellen verteilten Daten.
- Help-View (siehe Abschnitt 3.8.4): Help-Views dienen als Selektionsmethode in Suchhilfen.

3.8.1 Datenbank-View

Wenn Sie in dem Pop-up-Fenster zur Anlage eines Views den Typ **Datenbank-View** ausgewählt haben, gelangen Sie in den Datenbank-View-Editor:

1. Hier geben Sie auf der Registerkarte **Tabellen/Joinbedingungen** die zu verknüpfenden Tabellen an. In Open SQL wäre dies die `JOIN`-Klausel. Die Tabellen müssen dabei über ihre Primär- und Fremdschlüssel miteinander verknüpfbar sein. Tragen Sie die Tabellen unter **Tabellen** auf der linken Seite ein (siehe Abbildung 3.26), und klicken Sie anschließend auf den Button **Beziehungen** unterhalb der Tabelle. Dadurch werden Ihnen bereits mögliche Verknüpfungen vorgeschlagen, von denen Sie eine auswählen können.

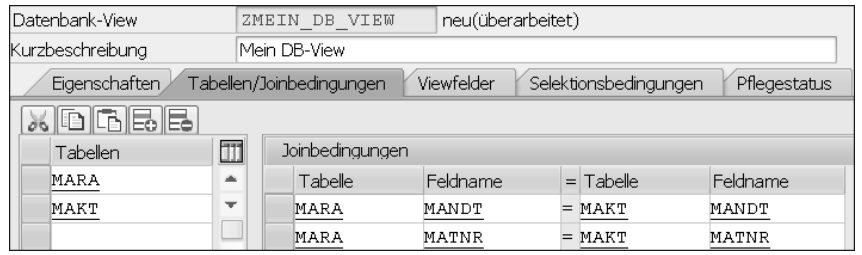

Abbildung 3.26 Join-Bedingung eines Views

2. Nun wählen Sie auf der Registerkarte **Viewfelder** (siehe Abbildung 3.27) die gewünschten Felder der ausgewählten Tabellen aus. Bei einer Definition in Open SQL entspräche das der Feldleiste nach der `SELECT`-Anweisung. Die View-Felder können Sie entweder manuell in die Tabelle eintragen oder sie über den Button **Tabellenfelder** automatisch hinzufügen lassen.

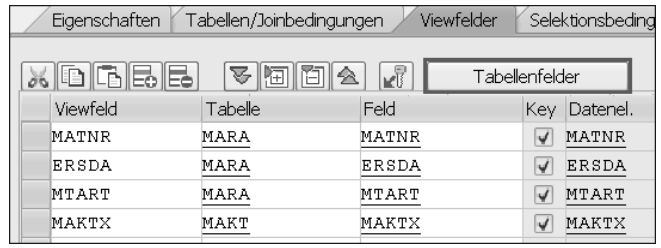

Abbildung 3.27 Felder eines Views

3. Jetzt können Sie Ihren View **Aktivieren** () und testen.

Sie haben darüber hinaus aber noch die Möglichkeit, auf der Registerkarte **Selektionsbedingungen** (siehe Abbildung 3.28) eine Selektionsbedingung analog zur einer `WHERE`-Klausel in einer Open-SQL-Anweisung anzugeben, um so nur eine Teilmenge der Daten zu selektieren. Auch dabei können Sie die Felder wieder über den Button **Tabellenfelder** automatisch hinzufügen lassen. Die möglichen Operatoren und Vergleichswerte entsprechen dabei genau der Schreibweise einer Open-SQL-Anweisung.

Für die Operatoren steht als Unterstützung eine F4-Hilfe bereit. Über die Spalte **AND/OR** können Sie Ihre Bedingungen verknüpfen.

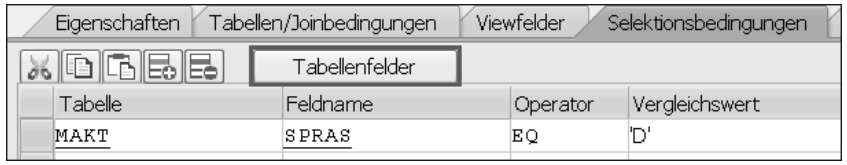

Abbildung 3.28 Selektionsbedingung eines Views

Aktivieren (Strg+F3) Sie gegebenenfalls erneut. Nun können Sie Ihren View testen. Dies ist über den Button **Datenanzeige** (F8) in der Funktionsleiste oder – wie eingangs beschrieben – über eine Open-SQL-Anweisung möglich.

Wie Sie in Listing 3.1 sehen können, können auch hier nochmals eine Selektion sowie eine Projektion vorgenommen werden. Die Selektion von Datenbanktabellen mit ABAP ist in Kapitel 9, »Zugriff auf Datenbanken«, beschrieben.

```
DATA: lt_mat_texte TYPE TABLE OF zmein_db_view.
SELECT matnr mtktx FROM zmein_db_view
   INTO CORRESPONDING FIELDS OF TABLE lt_mat_texte
   WHERE ersda = sy-datum.
```

Listing 3.1 Verwendung eines Views in einer SELECT-Anweisung

3.8.2 Projektions-View

Wenn Sie im Pop-up-Fenster zum Anlegen eines Views den Typ **Projektions-View** ausgewählt haben, gelangen Sie zum Projektions-View-Editor. Hier haben Sie im Vergleich zum Datenbank-View-Editor nicht die Möglichkeit, Tabellen miteinander zu verknüpfen, sondern können lediglich auf Basis einer Tabelle eine *Projektion*, also eine Auswahl von Feldern, vornehmen:

1. Dazu tragen Sie auf der Registerkarte **Viewfelder**, wie in Abbildung 3.29 dargestellt, die Basistabelle ein und wählen die gewünschten Felder aus. Der Button **Tabellenfelder** hilft Ihnen wieder bei der Auswahl der Felder.

2. Darüber hinaus können Sie auf der Registerkarte **Pflegestatus** steuern, ob auf die Tabelle nur lesend oder auch schreibend zugegriffen werden darf. Für die Auswahl der Felder für die **Data Browser/Tabellensicht-Pflege** gelten dieselben Erläuterungen wie bei der Anlage einer Tabelle (siehe Abschnitt 3.5.4, »Tabellensicht-Pflege«).

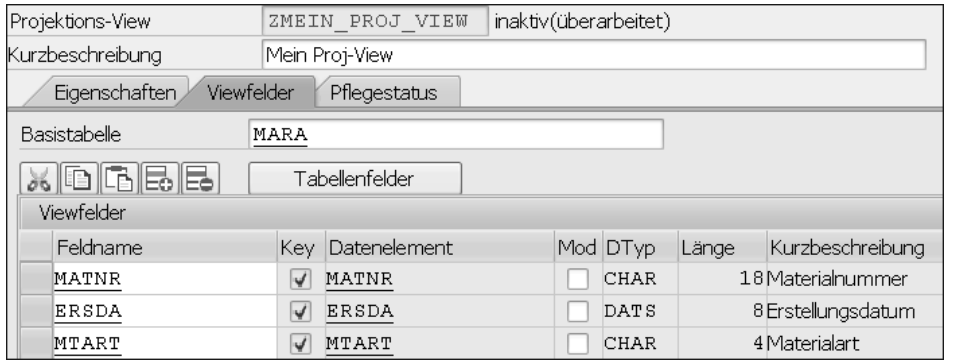

Abbildung 3.29 View-Felder eines Projektions-Views

3.8.3 Pflege-View

Wenn Sie in dem Pop-up-Fenster zum Anlegen eines Views den Typ **Pflege-View** ausgewählt haben, gelangen Sie in den Pflege-View-Editor. Ihnen wird vermutlich auffallen, dass der Pflege-View-Editor genau denselben Aufbau hat wie der für Datenbank-Views (siehe Abschnitt 3.8.1).

Der Unterschied ist, dass Sie hier auf der Registerkarte **Tabellen/Joinbedingungen** gezwungen sind, bei der Definition von Tabellenverknüpfungen über den Button **Beziehungen** unterhalb der linken Tabelle zu arbeiten, die Tabellen also nicht frei eintragen können. Hiermit soll sichergestellt werden, dass nur mit Fremdschlüsselbeziehungen gearbeitet wird. Hintergrund ist, dass ein Pflege-View die Pflege mehrerer verknüpfter Tabellen ermöglicht. Sie können damit im Unterschied zu normalen Datenbank-Views auch ändernde Open-SQL-Befehle anwenden bzw. einen Pflegedialog generieren (siehe Abschnitt 3.9).

Um den Pflege-View anzulegen, gehen Sie wie folgt vor:

1. Tragen Sie, wie in Abschnitt 3.8.1, »Datenbank-View«, erläutert, die Basistabelle unter **Tabellen** ein, und verknüpfen Sie diese über den Button **Beziehungen**.
2. Anschließend können Sie auf der Registerkarte **Pflegestatus** auswählen, inwiefern Ihr Pflege-View bearbeitet werden darf (siehe Abbildung 3.30). Für die Auswahlmöglichkeiten **Auslieferungsklasse** und **Data Browser/Tabellensicht-Pflege** gelten dabei dieselben Erläuterungen wie bei der Anlage einer Datenbanktabelle (siehe Abschnitt 3.5.2 und Abschnitt 3.5.4, »Tabellensicht-Pflege«).

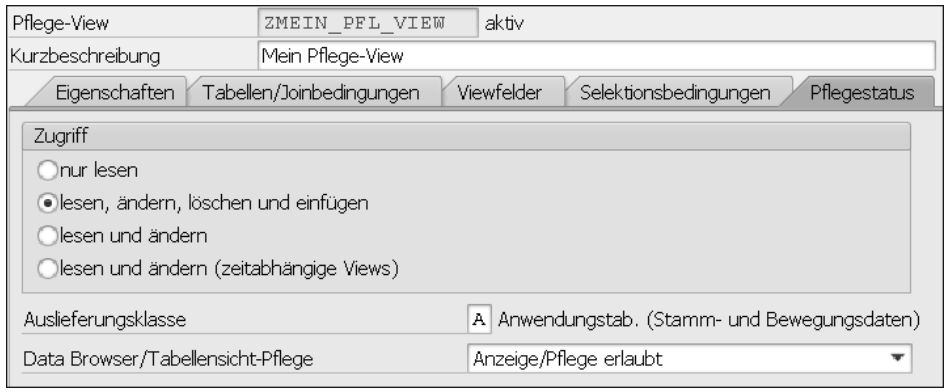

Abbildung 3.30 Pflegestatus eines Pflege-Views definieren

3.8.4 Help-View

Wenn Sie im Pop-up-Fenster zum Anlegen eines Views den Typ **Help-View** ausgewählt haben, gelangen Sie in den Help-View-Editor. Dieser ist wie der Editor zur Anlage eines Pflege-Views aufgebaut. Auch hier müssen Sie die einzelnen Tabellenverknüpfungen auf der Registerkarte **Tabellen/Joinbedingungen** über den Button **Beziehungen** eintragen.

Anschließend können Sie diesen Help-View als Datenquelle für eine Suchhilfe angeben. Dieses Vorgehen ist in Abschnitt 3.10, »Suchhilfen«, beschrieben.

3.9 Pflegedialoge

Datenbanktabellen müssen mit Daten gefüllt werden. Damit dies nicht ausschließlich über SQL-Befehle (siehe Kapitel 9, »Zugriff auf Datenbanken«) geschehen kann, bietet SAP die Möglichkeit, einen sogenannten *Pflegedialog* für eine Datenbanktabelle anzulegen. Über einen solchen Pflegedialog können Sie den Inhalt einer Datenbanktabelle pflegen. Sie können Zeilen hinzufügen, löschen oder ändern. Häufig wird für Pflegedialoge auch der Begriff *SM30-Tabelle* verwendet, da diese über den Transaktionscode SM30 (Tabellensicht-Pflege) zugänglich sind.

In den folgenden Abschnitten zeige ich Ihnen, wie Sie einen Pflegedialog anlegen (siehe Abschnitt 3.9.1) und die Eingabemaske verbreitern können (siehe Abschnitt 3.9.2).

3.9.1 Pflegedialog anlegen

Zur Anlage eines Pflegedialogs gehen Sie wie folgt vor:

1. Wechseln Sie in Transaktion SE11, und lassen Sie sich die Tabelle anzeigen, zu der Sie den Pflegedialog anlegen wollen (siehe Abschnitt 3.5.1, »Datenbanktabellen anzeigen«).
2. Aus der Tabellenanzeige können Sie nun über den Pfad **Hilfsmittel • Tabellenpflegegenerator** im Hauptmenü zur Anlage eines Pflegedialogs gelangen.
3. Folgende Angaben müssen Sie in der Anlageoberfläche machen (siehe Abbildung 3.31).

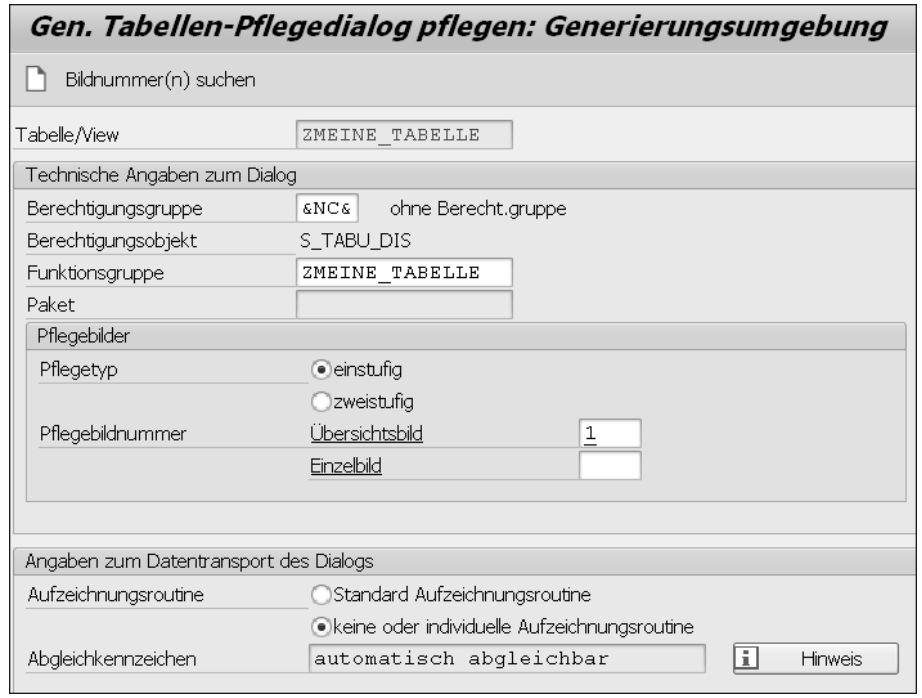

Abbildung 3.31 Generierung eines Pflegedialogs

- **Berechtigungsgruppe:** Wer darf später auf die Tabelle zugreifen, alle Benutzer oder nur bestimmte?
- **Funktionsgruppe:** die Funktionsgruppe, in der die Dynpros und die Ablauflogik für den Pflegedialog generiert werden sollen
- **Pflegetyp:** Möchten Sie eine Suchmaske als Einstiegsbild haben (**zweistufig**), oder soll der Benutzer direkt zur Tabellenanzeige gelangen (**einstufig**)?
- **Pflegebildnummer:** Wenn Sie als **Pflegetyp einstufig** ausgewählt haben, müssen Sie nur das Eingabefeld **Übersichtsbild** füllen, bei einem zweistufigen Pflegedialog müssen Sie sowohl das Eingabefeld **Übersichtsbild** als auch das Feld **Einzelbild** füllen. Vergeben Sie hier jeweils eine Nummer (z. B. 1 oder 2), diese müssen sich lediglich unterscheiden.

– **Aufzeichnungsroutine:** Wählen Sie **Standard Aufzeichnungsroutine**, fordert das System nach einer Pflege von Tabellenzeilen einen Transportauftrag an, damit ein Transport in andere Systeme ermöglicht wird. Bei der Auslieferungsklasse »C« ist diese Option standardmäßig eingestellt.

4. Klicken Sie anschließend in der Funktionsleiste auf **Anlegen** ( ).

Wenn alles geklappt hat, können Sie Ihren Pflegedialog nun in Transaktion SM30 testen. Geben Sie hierzu den Namen der Tabelle ein, und klicken Sie auf **Pflegen**.

3.9.2 Pflegedialog verbreitern

Wenn Sie Ihren Pflegedialog in Transaktion SM30 aufrufen, werden Sie feststellen, dass die Breite der Tabelle viel zu knapp bemessen ist. Die in Abbildung 3.32 markierte Fläche ist damit ungenutzt. Dies ist gerade bei vielen Spalten für die Anwender sehr unschön und umständlich. Daher gibt es einen Trick, um den Pflegedialog zu verbreitern.

Abbildung 3.32 Ungenutzter Platz eines Pflegedialogs

Weitere Informationen zur Dynpro-Programmierung

Zum Verständnis der folgenden Erläuterungen ist es hilfreich, wenn Sie bereits grundlegende Kenntnisse in der Dynpro-Programmierung haben. Die Grundlagen lernen Sie in Kapitel 14.

Die Vorgehensweise ist überschaubar:

1. Öffnen Sie die Funktionsgruppe zum Pflegedialog, indem Sie in Transaktion SE80 die Dropdown-Liste **Funktionsgruppe** nutzen.
2. Öffnen Sie das Dynpro für das Übersichtsbild des Pflegedialogs (im Ordner **Dynpros**, z. B. Dynpro 0001).
3. Wechseln Sie in den Bearbeitungsmodus über den Button **Anzeigen <-> Ändern** ( ).
4. Öffnen Sie den Layout-Editor über den Button **Layout**.

5. Verbreitern Sie, wie in Abbildung 3.33 dargestellt, den Hauptbereich für das Fenster, indem Sie den Rahmen mit der Maus nach rechts ziehen.

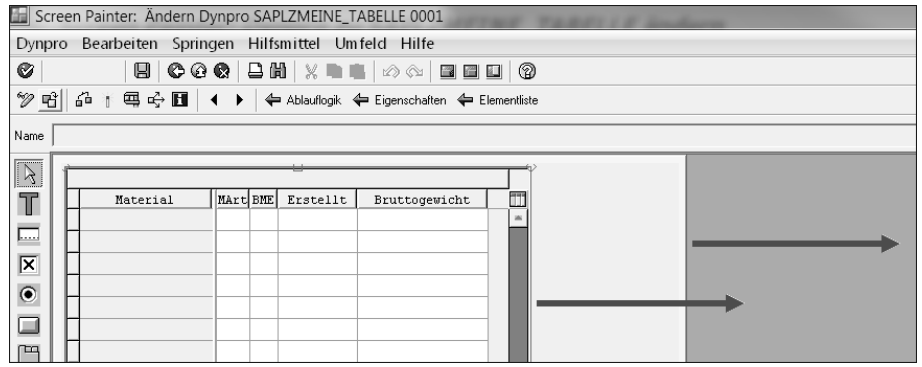

Abbildung 3.33 Verbreiterung eines Pflegedialogs im Screen Painter

Dynpro-Größe ändern

Falls Ihnen dabei die Meldung »Element(e) außerhalb der neuen Grenzen (Dynpro-Größe nicht verändert)« angezeigt wird, müssen Sie die Dynpro-Größe vergrößern. Scrollen Sie dazu, wie in Abbildung 3.34 dargestellt, zum unteren Rand des Bildschirms im Screen Painter, und ziehen Sie die Ecke des Dynpros mit gedrückter linker Maustaste erst nach unten und dann nach rechts.

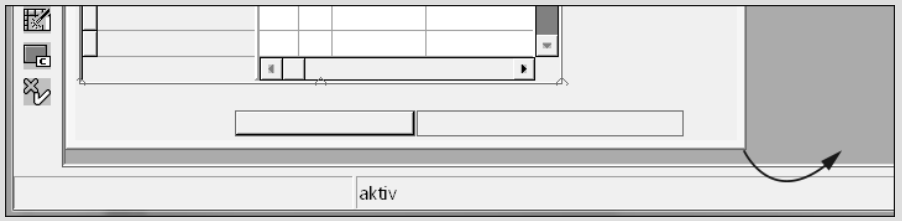

Abbildung 3.34 Dynpros im Screen Painter verbreitern

6. Aktivieren ( ) Sie anschließend, und testen Sie Ihre Änderungen in Transaktion SM30.

Neugenerierung bei Anpassung

Falls Sie den Pflegedialog später in Transaktion SE11 bzw. SE54 nochmals anpassen, wird das Dynpro teilweise neu generiert und wieder auf die Standardgröße gebracht. In diesem Fall müssen Sie also die beschriebenen Änderungen noch einmal durchführen.

3.10 Suchhilfen

Suchhilfen stehen innerhalb des SAP-Systems als Eingabehilfen für den Anwender bereit. Sie zeigen dem Anwender eine Liste aller möglichen Eingabewerte für ein Eingabefeld. Die möglichen Eingabewerte können, wie in Abbildung 3.35 zu erkennen, durch weitere Informationen angereichert sein, damit der Anwender z. B. zu einer Materialnummer auch den zugehörigen Materialtext sieht und ihm so die Auswahl leichter fällt. Durch einen Doppelklick auf eine Zeile in der Suchhilfe wird der ausgewählte Wert in das Eingabefeld übernommen.

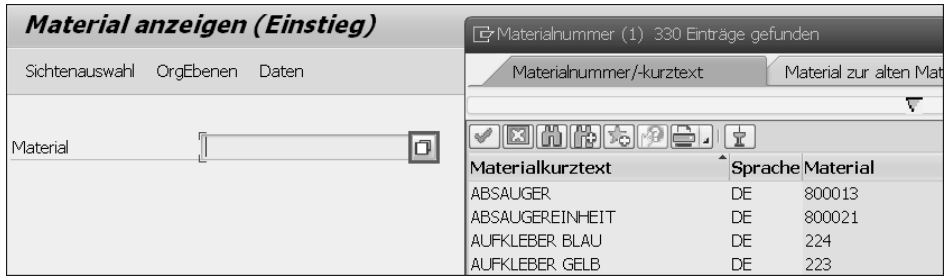

Abbildung 3.35 Eine Suchhilfe für die Materialnummer

Innerhalb des ABAP Dictionarys können Sie eigene Suchhilfen anlegen. Geben Sie dazu den gewünschten Namen der Suchhilfe auf dem Hauptübersichtsbildschirm der Transaktion SE11 im Feld **Suchhilfe** ein, und klicken Sie auf **Anlegen**. Im sich öffnenden Pop-up-Fenster haben Sie folgende Arten von Suchhilfen zur Auswahl:

- *Elementare Suchhilfe* als Standardeingabehilfe (siehe Abschnitt 3.10.1)
- *Sammelsuchhilfe* als Zusammenfassung von mehreren elementaren Suchhilfen, wobei für jede Suchhilfe eine Registerkarte angezeigt wird (siehe Abschnitt 3.10.2)

Innerhalb von Suchhilfen bietet Ihnen das System darüber hinaus mit *Append-Suchhilfen* die Möglichkeit, Sammelsuchhilfen modifikationsfrei zu erweitern. Dieses Vorgehen ist in Abschnitt 21.4.8, »Suchhilfenerweiterungen«, beschrieben.

3.10.1 Elementare Suchhilfe

Wenn Sie im Pop-up-Fenster zur Anlage einer Suchhilfe den Typ **Elementare Suchhilfe** ausgewählt haben, gelangen Sie in den Suchhilfen-Editor:

1. Zur Definition einer Suchhilfe müssen Sie zunächst folgende Eigenschaften definieren (siehe Abbildung 3.36):
   - **Selektionsmethode:** Wählen Sie, ob die Daten von einer Datenbank oder einem View gelesen werden sollen.

   - **Dialogverhalten:** Wählen Sie, ob ein Fenster zur Werteeinschränkung angezeigt werden soll oder ob die Ergebnisliste direkt angezeigt werden soll.

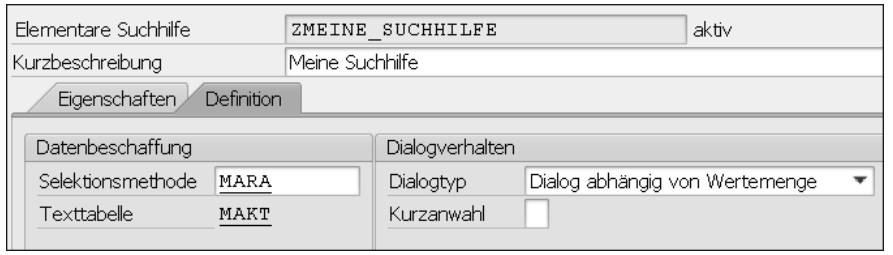

Abbildung 3.36 Selektionsmethode und Dialogverhalten einer Suchhilfe

2. Darüber hinaus müssen Sie die **Suchhilfeparameter** bestimmen (siehe Abbildung 3.37). Das sind die Felder, die im Suchhilfefenster angezeigt werden sollen. Dabei müssen Sie auch bestimmen, welche Felder bei der Auswahl eines Eintrags aus der Ergebnisliste übernommen werden.

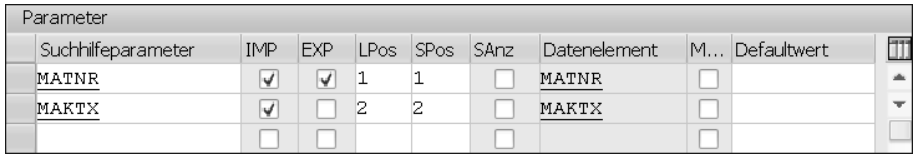

Abbildung 3.37 Definition von Suchhilfeparametern

3. Über die Spalte **IMP** definieren Sie, welche Werte aus Ihrer Datenquelle übernommen werden, also in der Wertetabelle angezeigt werden könnten.
4. Über die Spalte **EXP** definieren Sie, welche Werte in die Maske übernommen werden.
5. Über die Spalte **LPos** definieren Sie, ob das Feld in der Trefferliste auch wirklich dargestellt werden soll. Mit der Nummer geben Sie die Reihenfolge der Spalten an (z. B. zuerst das Feld `MATNR` für das **Material** ❶ und anschließend das Feld `MAKTX` für den **Materialkurztext** ❷, siehe Abbildung 3.38). Soll der Parameter nicht auf der Trefferliste erscheinen, lassen Sie dieses Feld frei.
6. Über die Spalte **SPos** definieren Sie, ob das Feld in der Suchmaske dargestellt werden soll. Mit der Nummer geben Sie die Reihenfolge der Eingabefelder an (z. B. erst das Feld `MATNR` für das **Material** ❸ und dann das Feld `MAKTX` für den **Materialkurztext** ❹, siehe Abbildung 3.38). Soll der Parameter nicht auf der Suchmaske erscheinen, so lassen Sie dieses Feld frei.
7. Zum Schluss **Aktivieren** (Strg+F3) Sie Ihre Suchhilfe.

Abbildung 3.38 Unterschied zwischen den Angaben in den Spalten »LPos« und »SPos«

Nun können Sie die Suchhilfe an ein Datenelement hängen (siehe auch Abschnitt 3.2) oder sie direkt in einem Selektionsbildschirm durch die Angabe eines Parameters mit dem Zusatz `MATCHCODE OBJECT` verwenden:

```
PARAMETERS: p_matnr TYPE matnr MATCHCODE OBJECT zmeine_suchhilfe.
```

3.10.2 Sammelsuchhilfe

Wenn Sie im Pop-up-Fenster zur Anlage einer Suchhilfe den Typ **Sammelsuchhilfe** ausgewählt haben, gelangen Sie zum Suchhilfen-Editor:

1. Zur Definition einer Sammelsuchhilfe müssen Sie zuerst die Suchhilfeparameter definieren (d. h. bestimmen, welche Werte übernommen und welche zurückgegeben werden). Diese Suchhilfeparameter tragen Sie auf der Registerkarte **Definition** ein (siehe Abbildung 3.39).

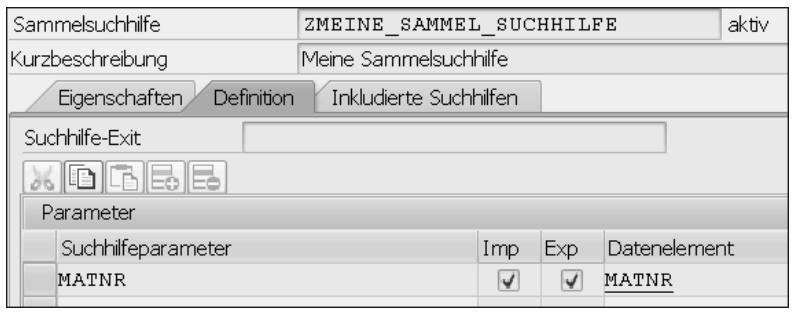

Abbildung 3.39 Suchhilfeparameter einer Sammelsuchhilfe

2. Anschließend geben Sie auf der Registerkarte **Inkludierte Suchhilfen** die elementaren Suchhilfen an, die zur Sammelsuchhilfe zusammengefasst werden sollen.
3. Nun müssen Sie die auf der Registerkarte **Definition** definierten Suchhilfeparameter den Suchhilfeparametern der inkludierten Suchhilfen zuordnen. Klicken Sie

hierzu, wie in Abbildung 3.40 dargestellt, auf die Zeile der Suchhilfe ❶ und anschließend auf den Button **Parameterzuordnung** ❷.

4. Tragen Sie einen **Bezugsparameter** ein, und bestätigen Sie Ihre Zuordnung mit dem Button **Übernehmen**.

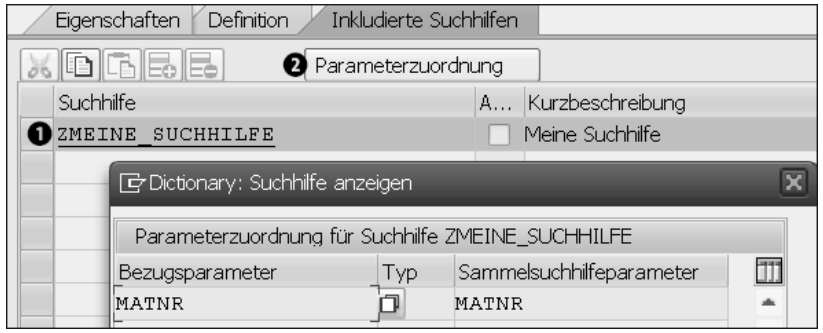

Abbildung 3.40 Parameterzuordnung in einer Sammelsuchhilfe

5. **Aktivieren** (Strg+F3) Sie anschließend Ihre Sammelsuchhilfe.

Nun können Sie die Sammelsuchhilfe an ein Datenelement hängen (siehe Abschnitt 3.2) oder wie die elementare Suchhilfe in einem Selektionsbildschirm verwenden (siehe Abschnitt 3.10.1).

3.11 Datenbank-Utility-Tool

Das Datenbank-Utility-Tool brauchen Sie vor allem dann, wenn Ihre Datenbanktabelle bereits Daten beinhaltet, Sie aber aufgrund einer neuen Anforderung z. B. neue Schlüsselfelder hinzufügen müssen und beim Aktivieren der Tabelle dadurch folgende Fehlermeldung erhalten: »Strukturänderung auf Feldebene (Bitte Tabelle umsetzen).«

Wählen Sie im Hauptmenü **Hilfsmittel • Datenbankobjekt • Datenbank-Utility**, um das Tool zu öffnen. Sie können nun drei grundlegende Datenbankoperationen (siehe Abbildung 3.41) durchführen:

- **Datenbanktabelle anlegen**
- **Datenbanktabelle löschen**
- **Aktivieren und Datenbank anpassen**

Um die geänderte Datenbanktabelle zu aktivieren, wählen Sie **Aktivieren und Datenbank anpassen**. Bestätigen Sie das Pop-up-Fenster **Auftrag: Anpassen** mit **Ja**.

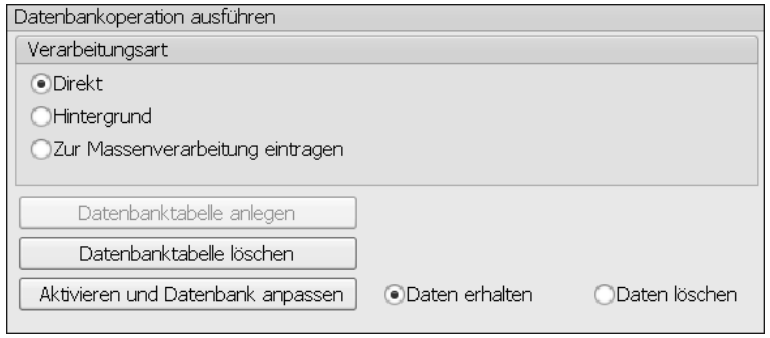

Abbildung 3.41 Datenbankoperationen mit dem Datenbank-Utility-Tool

3.12 Das Sperrkonzept

Über das Sperrkonzept soll verhindert werden, dass durch gleichzeitige Änderungen Datenschiefstände in einer Datenbanktabelle entstehen. Vergleichen könnte man diesen Mechanismus mit dem Sperrmechanismus in Microsoft Excel: Nur ein Benutzer darf mit einer Excel-Tabelle arbeiten, ein anderer Benutzer darf derweil auf die Daten nur lesend zugreifen.

Das Sperrkonzept im ABAP Dictionary geht hier noch einen Schritt weiter und ermöglicht nicht nur das Sperren einer kompletten Datenbanktabelle, sondern auch das Sperren einzelner Einträge einer Datenbanktabelle. In einem Sperrobjekt werden die Tabellen mit ihren Schlüsselfeldern angegeben, in denen bestimmte Datensätze gesperrt werden sollen. Das Setzen bzw. Freigeben von Sperren erfolgt durch den Aufruf von Funktionsbausteinen, die automatisch bei der Definition eines Sperrobjekts generiert werden.

Ein Sperrobjekt anlegen

Um ein Sperrobjekt anzulegen, gehen Sie wie folgt vor:

1. Geben Sie den gewünschten Namen des Sperrobjekts auf dem Hauptübersichtsbildschirm der Transaktion SE11 im Feld **Sperrobjekt** ein. Der Name muss dabei mit einem `E` anfangen, gefolgt vom Kundennamensraum (z. B. `Z` oder `Y`). Beispielsweise könnte Ihr Sperrobjekt `EZ_MEIN_SPERROBJ` heißen.
2. Klicken Sie anschließend auf **Anlegen**.
3. Es öffnet sich der Sperrobjekt-Editor, in dem Sie nun auf der Registerkarte **Tabellen** die zu sperrende Tabelle eintragen und einen Sperrmodus auswählen (siehe Abbildung 3.42). Folgende (wichtige) Sperrmodi stehen Ihnen dazu zur Verfügung:
   - **Schreibsperre:** Nur ein Benutzer kann die gesperrten Daten lesen bzw. bearbeiten.

   - **Lesesperre:** Mehrere Benutzer können auf dieselben Daten (lesend) zugreifen. Sobald ein Benutzer jedoch die Daten bearbeitet, hat ein zweiter Benutzer keinen Zugang auf diese Daten.
   - **erweiterte Schreibsperre:** Eine Transaktion kann nur eine Sperre anfragen, jede weitere Anforderung einer Sperre wird abgewiesen.

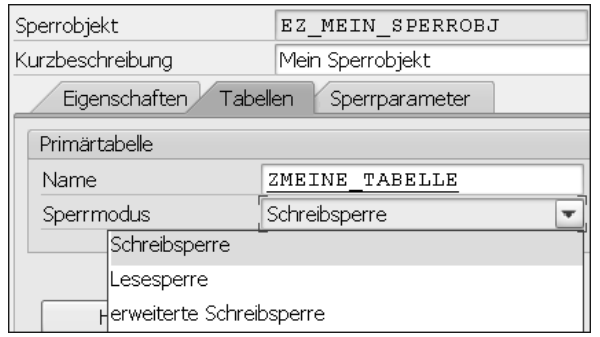

Abbildung 3.42 Sperrmodi eines Sperrobjekts

4. Auf der Registerkarte **Sperrparameter** können Sie zusätzlich die Sperrparameter (in Form der Primärschlüssel) bearbeiten und gegebenenfalls einen Primärschlüssel deaktivieren.
5. Zuletzt **Aktivieren** (Strg+F3) Sie Ihr Sperrobjekt.

Wenn Sie nun im Hauptmenü den Pfad **Springen • Sperrbausteine** wählen, werden Ihnen zwei Funktionsbausteine präsentiert (siehe Abbildung 3.43).

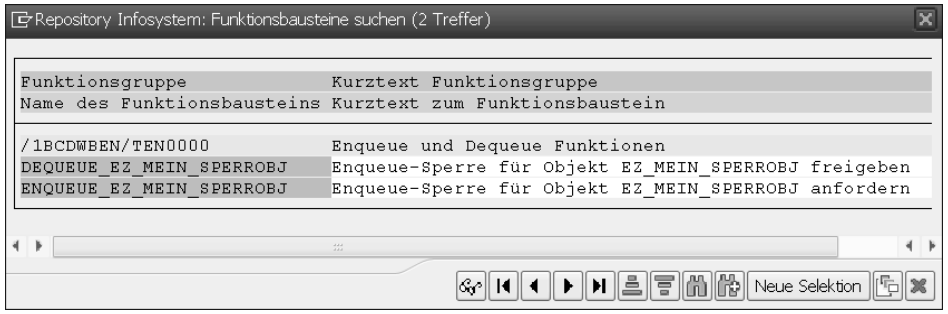

Abbildung 3.43 Sperrbausteine eines Sperrobjekts

Diese können Sie nun verwenden, um die Datenbanktabelle zu sperren bzw. zu entsperren. Wie Sie Funktionsbausteine mit ABAP generell aufrufen, ist in Abschnitt 13.4.2, »Aufruf von Funktionsbausteinen«, beschrieben. Sie müssen beim Aufruf lediglich die Primärschlüssel der zu sperrenden Datensätze angeben.

Sperreinträge überprüfen

Über die Transaktion SM12 können Sie sich alle Sperreinträge für das gesamte System, einen Mandanten, einen Benutzer oder eine Datenbanktabelle anzeigen lassen. Hiermit haben Sie die Möglichkeit, zu überprüfen, ob das Setzen einer Sperre mit den obigen Sperrbausteinen funktioniert hat.

Einleitung

ABAP ist die von SAP entwickelte Programmiersprache für die Erstellung kundenspezifischer Anwendungsprogramme. Die Abkürzung steht für Advanced Business Application Programming. Darüber hinaus nutzt SAP selbst ABAP auch als Basis für SAP ERP, die SAP Business Suite und die SAP-NetWeaver-Plattform. All diese Systeme basieren auf Millionen von in ABAP geschriebenen Zeilen.

Die Abkürzung ABAP bedeutete ursprünglich Allgemeiner Berichtsaufbereitungsprozessor und war, wie es der Name vermuten lässt, dafür gedacht, kundeneigene Berichte aufzubereiten. ABAP orientierte sich in seiner ursprünglichen Form an der Syntax der Programmiersprache COBOL. Über die Jahrzehnte wurde ABAP kontinuierlich weiterentwickelt und verbessert. So hat sich der Sprachumfang mittlerweile vervielfacht, da immer wieder neue Anweisungen und Funktionen hinzukamen, wobei ältere aus Kompatibilitätsgründen beibehalten wurden.

Die aktuelle Version von ABAP (7.54, Stand Ende 2020) steht in ihrer Funktionalität anderen modernen Programmiersprachen kaum noch nach. Bereits 2004 wurde der Sprachumfang um die objektorientierten Elemente von *ABAP Objects* erweitert. Dennoch findet man aufgrund der erwähnten Kompatibilitätssicherung in ABAP auch heute noch viel ältere Funktionalität, die sich mit den neuen, modernen Konstrukten vermischt hat. In der Praxis hat es der Entwickler daher häufig mit einer Mischform aus »alter« und »neuer« Welt zu tun. Eine einseitige Betrachtung ergibt daher keinen Sinn.

Dennoch ist die Zukunft von ABAP bereits angebrochen. ABAP hat durch die letzten beiden Releases 7.40 und 7.50/7.51 sehr viele neue Funktionen gewonnen, die in die Kapitel dieses Buches integriert sind. In Abbildung 1 können Sie die verschiedenen ABAP-Versionen über die Jahre dargestellt sehen.

Auffällig ist hier, dass zwischen 2004 und 2013 praktisch keine Neuerungen in ABAP eingeführt wurden. Dies war auch die Zeit, als es hieß, dass Java der Nachfolger von ABAP werden sollte. Dies hat sich aber zum Glück nicht durchgesetzt.

Besonders wichtig in Bezug auf dieses neue ABAP sind die neue Datenbank SAP HANA und die neue Oberflächentechnologie SAPUI5 bzw. die von SAP bereitgestellten SAP-Fiori-Apps. Diese neuen Technologien sind das Aushängeschild des SAP-ERP-Nachfolgers SAP S/4HANA. Doch auch wenn die SAP Business Suite für SAP S/4HANA (auf Vorlage der alten Systeme) komplett überarbeitet wurde, bleibt ABAP die Kernprogrammiersprache auch für dieses System.

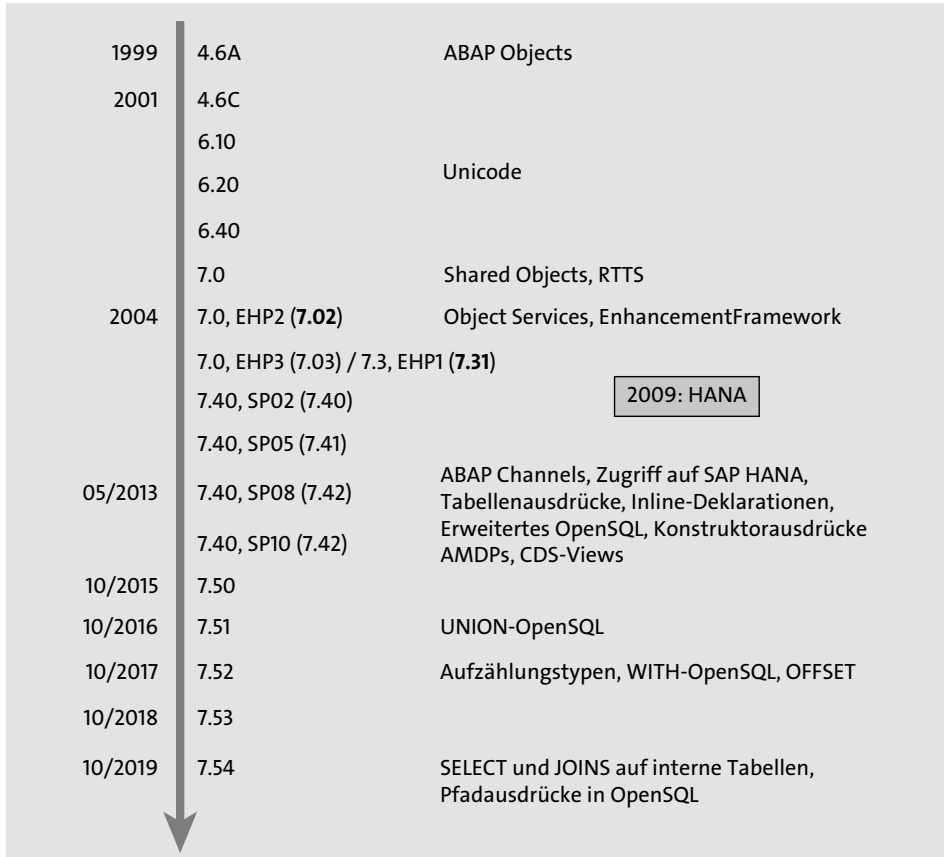

Abbildung 1 Neuerung in ABAP seit 1999 bis heute

Altlasten existieren weiterhin

Es ist wichtig zu verstehen, dass in der On-Premise-Version von SAP S/4HANA weiterhin alle alten Sprachelemente vorzufinden sind. Das heißt, Sie können auch hier Makros, Unterprogramme, Funktionsbausteine etc. verwenden. Technische gesehen ist es für uns ABAP-Entwickler der gleiche Unterbau. In anderen Worten: Es muss alles beherrscht werden, insbesondere, wenn Altanwendungen gewartet oder erweitert werden sollen.

Betrachtet man die durchschnittliche Halbwertszeit (d. h. die Einsatzzeit beim Kunden) eines ERP-Systems von ca. 20 Jahren (SAP R/3 kam 1993, SAP ECC 2004), wird uns ABAP durch SAP S/4HANA noch lange erhalten bleiben – auch wenn dies in den letzten Jahren (z. B. im Rahmen der Ankündigung der Java-Strategie) nicht immer als Selbstverständlichkeit galt.

In anderen Worten: ABAP ist die Programmiersprache von SAP und wird dies bleiben. Herzlich willkommen seien daher alle Leser, die sich entschieden haben, sich intensiver mit ABAP im Allgemeinen und ABAP Objects im Speziellen zu beschäftigen.

Aufbau dieses Buches

Zuallererst möchte ich Ihnen einen Überblick über das vor Ihnen liegende Buch geben. Im Anschluss habe ich Ihnen verschiedene Lesepläne zusammengestellt – abhängig davon, welche Vorkenntnisse Sie mitbringen.

- **Teil I, »Die Werkzeugkiste des ABAP-Entwicklers«**, beschäftigt sich mit den Entwicklungswerkzeugen des SAP GUI wie der eingebauten ABAP Workbench (Transaktion SE80). Außerdem stelle ich Ihnen hier die neuen ABAP Development Tools for SAP NetWeaver (ADT) für die Entwicklungsumgebung Eclipse vor.
- **Teil II, »Der Kern der Sprache ABAP«**, beschäftigt sich mit dem ABAP-Sprachkern. Von absolut grundlegenden Anweisungen über den Zugriff auf Datenbanktabellen, die Verarbeitung der Daten und ABAP-Objects-Anweisungen bis hin zur Report- und Dynpro-Programmierung ist hier alles vertreten. So hat hier auch der Zugriff auf die neue SAP-HANA-Datenbank mit ABAP (auch *ABAP für HANA* genannt) mit Eingang gefunden. Insbesondere finden Sie in diesem Teil die mit den Releases 7.40 und 7.50/1 neu eingeführten Sprachbefehle und Funktionen.
- **Teil III, »Techniken zur Qualitätssicherung«**, dreht sich um den Test und die Qualitätskontrolle Ihrer geschriebenen Anweisungen. Hier werden die gezielte Verwendung des ABAP Debuggers sowie die zur Verfügung stehenden Analysewerkzeuge erläutert.
- **Teil IV, »Fortgeschrittene Programmiertechniken«**, behandelt weitere Themen aus dem Umfeld von ABAP. Dazu gehören die Entwicklung von Schnittstellen, die Erweiterung von SAP-Standardprogrammen, die Entwicklung von Formularen und Tabellenanzeigen sowie auch fortgeschrittene Techniken wie die dynamische Programmierung oder die Anwendung objektorientierter Frameworks wie der Object Services oder Shared Objects.
- **Teil V, »Objektorientierte Programmierung«**, vermittelt in erster Linie die Grundlagen der objektorientierten Programmierung mit ABAP und zeigt konkrete Methoden für den Entwicklungsprozess. Insbesondere werden hier die objektorientierte Modellierung sowie Hilfsmittel wie UML und Entwurfsmuster besprochen.
- Im letzten Teil des Buches, **Teil VI, »Ein Blick über den Tellerrand: Was Sie als ABAP-Entwickler sonst noch kennen sollten«**, möchte ich Ihnen den Grund für die vielen Änderungen der letzten Releases etwas genauer vorstellen: SAP HANA als neue Datenbankplattform sowie SAPUI5 als neue Web-Frontend-Technologie, die eine regelrechte Innovationswelle ab 2010 angestoßen haben. In diesem Teil

gehe ich insbesondere sowohl auf die vielen neuen Begriffe im Umfeld dieser Technologien als auch auf die Entwicklung von OData-Services ein.

Der Anhang fasst alle wichtigen ABAP-Anweisungen, obsoleten ABAP-Anweisungen, Systemfelder, eingebauten Datentypen, nützliche Funktionsbausteine und Klassen, Transaktionscodes sowie technische Tabellen zur Programmierung mit ABAP zusammen.

Ihr Weg durch das Buch

Da sich dieses Buch an mehrere Zielgruppen richtet (ABAP-Einsteiger, -Kenner und -Experten) und gleichermaßen als Nachschlagewerk für den Alltag dienen soll, finden Sie im Folgenden einige Hinweise, wie Sie aus dem Buch für sich am meisten Gewinn erzielen können.

Aus folgenden Leseplänen können Sie wählen:

- Einsteiger in ABAP
- Einsteiger in ABAP Objects
- Kenner von ABAP Objects (vor allem der Releases vor und bis 7.31)
- Leser der Vorauflage
- Interessenten an den neuen Technologien
- Experten

Die Lesepläne bauen aufeinander auf, vom Einsteiger bis zum Experten. Versuchen Sie, wann immer es geht, so viel wie möglich am System nachzuvollziehen, da sich Programmierung am besten durch die konkrete Anwendung erlernen lässt.

In hervorgehobenen Informationskästen sind Inhalte zu finden, die wissenswert und hilfreich sind, aber etwas außerhalb der eigentlichen Erläuterung stehen. Damit Sie die Informationen in den Kästen sofort einordnen können, haben wir die Kästen mit Symbolen gekennzeichnet:

[+] Die mit diesem Symbol gekennzeichneten *Tipps* geben Ihnen spezielle Empfehlungen, die Ihnen die Arbeit erleichtern können.

[»] In Kästen, die mit diesem Symbol gekennzeichnet sind, finden Sie Informationen zu *weiterführenden Themen* oder wichtigen Inhalten, die Sie sich merken sollten.

[!] Dieses Symbol weist Sie auf *Besonderheiten* hin, die Sie beachten sollten. Es *warnt Sie* außerdem vor häufig gemachten Fehlern oder Problemen, die auftreten können.

Leseplan für Einsteiger in ABAP

Fangen Sie mit der Lektüre des Kapitel 1 an, um sich mit der Entwicklungsumgebung vertraut zu machen. Insbesondere ist hier Abschnitt 1.4.2, »Einen Report anlegen«,

von Bedeutung, der Ihnen zeigt, wie Sie einen neuen Report in Transaktion SE80 anlegen und ausführen können.

Bevor Sie sich dann mit den Anweisungen von ABAP auseinandersetzen, ist es wichtig, die Rolle des in Kapitel 3 beschriebenen ABAP Dictionarys zu verstehen. Lesen Sie sich hier alle Abschnitte bis Abschnitt 3.4, »Tabellentypen«, durch. Die folgenden Abschnitte können Sie erst mal außer Acht lassen.

Mit Ihrem ersten angelegten Report und dem Grundwissen über das ABAP Dictionary können Sie nun die in Kapitel 7, »Die ABAP-Grundbefehle«, beschriebenen grundlegenden ABAP-Anweisungen ausprobieren. Versuchen Sie, die vielen Beispiele direkt bei sich im System auszuprobieren. Überspringen Sie erst mal die als fortgeschritten markierten Abschnitte.

Nach einer kurzen Verschnaufpause können Sie sich direkt intensiv mit Kapitel 8 auseinandersetzen, das die internen Tabellen von ABAP intensiv behandelt. Kapitel 9 zeigt Ihnen im Anschluss, wie Sie Daten von der Datenbank in interne Tabellen laden und weiterverarbeiten können.

Lesen Sie nun die restlichen Abschnitte (ab Abschnitt 3.5) von Kapitel 3, »Das ABAP Dictionary«. Insbesondere ist hier Abschnitt 3.5, »Datenbanktabellen«, von Bedeutung, der Ihnen zeigt, wie Sie selbst eine Tabelle auf der Datenbank anlegen können. So eine Tabelle können Sie mit den in Kapitel 9, »Zugriff auf Datenbanken«, gelernten Open-SQL-Anweisungen bzw. den in Abschnitt 3.9 beschriebenen Pflegedialogen bearbeiten.

Haben Sie eine SAP-HANA-Datenbank zur Verfügung, empfehle ich Ihnen nun die Lektüre von Kapitel 30, »SAP HANA«, sowie anschließend von Kapitel 10, »Zugriff auf SAP-HANA-Entwicklungsobjekte«. Sollten Sie keine SAP-HANA-Datenbank zur Verfügung haben, so ist dies nicht weiter schlimm, da Sie diese beiden Kapitel jederzeit nachträglich ohne Probleme durcharbeiten können.

Kapitel 12 beschäftigt sich dann mit der Erstellung von Selektionsbildschirmen in Reports. Dies hilft Ihnen dabei, Eingabemasken für Ihre Anwender bereitzustellen. Kapitel 4 zeigt Ihnen, wie Sie es schaffen, Ihren Report mit einem Transaktionscode zu verknüpfen, damit Ihre Anwender den von Ihnen entwickelten Report direkt aufrufen können.

Da die Programme durch die vielen Anweisungen immer größer werden, finden Sie in Kapitel 13 die in ABAP zur Verfügung stehenden Strukturierungselemente, die Ihnen dabei helfen, Ihre Programme in kleinere, voneinander abgetrennte Bereiche zu unterteilen. Insbesondere ist hier Abschnitt 13.4, »Funktionsbausteine«, von Bedeutung. Lesen Sie sich dazu auch Kapitel 5, »Der Function Builder«, durch, um zu verstehen, wie Sie eigene Funktionsbausteine anlegen und verwenden können. Natürlich können Sie auch direkt Klassen und Interfaces zur objektorientieren Strukturierung Ihrer Programme nutzen, wie in Kapitel 11, »Die ABAP-Objects-Syntax«, beschrieben.

Anschließend können Sie dem Leseplan für Einsteiger in ABAP Objects folgen. Haben Sie bis hierhin alle Kapitel gemeistert, können Sie sich dem Umfeldwissen zu ABAP widmen, hierzu gehören die Kapitel in Teil III, Teil IV und Teil VI. Heben Sie sich Kapitel 24, »Fortgeschrittene Programmiertechniken«, bis ganz zum Schluss auf.

Um das große Ganze zwischen all den ABAP-Technologien zu verstehen, empfehle ich Ihnen außerdem Kapitel 29, »ABAP-Programmiermodelle«, wo ich Ihnen einen Überblick über die aktuelle und zukünftige Art der Programmierung mit ABAP erläutere.

Leseplan für Einsteiger in ABAP Objects

Dieser Leseplan ist für all jene gedacht, die sich mit dem Sprachkern von ABAP bestens auskennen, aber noch nicht die Chance hatten, sich mit ABAP Objects auseinanderzusetzen, oder bei denen der letzte Einsatz der objektorientierten Konzepte schon eine Weile her ist.

Lesen Sie zunächst Kapitel 25, »Grundlagen der Objektorientierung«, sowie Kapitel 11, »Die ABAP-Objects-Syntax«. Üben Sie nach der Lektüre dieser beiden Kapitel erst einmal intensiv die Anwendung von Klassen und Instanzen am Beispiel des SAP List Viewers (ALV) mithilfe von Kapitel 19. Konzentrieren Sie sich hier insbesondere auf die neue ALV-Anzeige auf Basis der Klasse `CL_SALV_TABLE`, die in Abschnitt 19.2 beschrieben ist.

Legen Sie, wie in Kapitel 6, »Der Class Builder«, beschrieben, eine eigene Klasse an, die Sie anschließend in Ihrem Report instanziieren und verwenden können. Hinweise zur objektorientierten Gestaltung von Reports gibt Abschnitt 28.6, »Objektorientierte Reports«.

Arbeiten Sie nun schrittweise die restlichen Kapitel in Teil V durch. Am Ende in der Einleitung zu Teil V finden Sie Verweise auf die konkrete Anwendung bereits in SAP existierender objektorientierter Frameworks. Dazu gehört beispielsweise das Business Object Processing Framework, das Sie in Kapitel 23 beschrieben finden.

Leseplan für Kenner von ABAP Objects

Sie kennen bereits ABAP und ABAP Objects und sind vor allem an den Neuerungen bis Release 7.54 interessiert? Dann ist dieser Leseplan richtig für Sie.

Wenn Sie die ABAP Development Tools (ADT) für Eclipse noch nicht kennen, sollten Sie direkt mit Kapitel 2 einsteigen. Anschließend sind für Sie insbesondere die Übersichten in den folgenden Abschnitten interessant, die Auskunft über die ABAP-Neuerungen geben und jeweils auf die Abschnitte im Buch verweisen, in denen diese ausführlicher behandelt werden:

- Abschnitt 7.15, »Änderungen und Neuerungen bis ABAP 7.54«
- Abschnitt 8.12, »Änderungen und Neuerungen im Umfeld von internen Tabellen bis ABAP 7.54«
- Abschnitt 9.9, »Änderungen und Neuerungen im Umfeld von Open SQL bis ABAP 7.54«
- Abschnitt 10.4, »Änderungen und Neuerungen beim Zugriff auf SAP-HANA-Entwicklungsobjekte bis ABAP 7.54«
- Abschnitt 11.15, »Änderungen und Neuerungen in ABAP Objects bis ABAP 7.54«

Insbesondere sollten Sie nach der Lektüre folgende Begriffe verstanden haben und anwenden können:

- Inline-Deklaration
- Tabellenausdrücke
- Änderungen an Open SQL, z. B. die neuen Open-SQL-Ausdrücke, eine Selektion mit `SELECT` auf interne Tabellen oder die Verwendung der neuen Anweisung `WITH`
- Konstruktorausdrücke, insbesondere die Verwendung der Zusätze von `BASE` und `LET`
- den Zusatz `GROUP BY` der Anweisung `LOOP AT`
- ABAP Managed Database Procedures (AMDP) zur Programmierung von Stored Procedures in Klassen (für SAP HANA)
- AMDP BAdIs
- ABAP-CDS-Views
- Mesh-Typen

Leser der Vorauflage

Sie kennen bereits die erste Auflage dieses Buches und sind nur am Delta von ABAP 7.31 und ABAP 7.54 interessiert?

Dann sind für Sie die folgenden Abschnitte interessant, die Auskunft über die ABAP-Neuerungen geben und jeweils auf die Abschnitte im Buch verweisen, in denen diese ausführlicher behandelt werden.

- Abschnitt 7.15, »Änderungen und Neuerungen bis ABAP 7.54«
- Abschnitt 8.12, »Änderungen und Neuerungen im Umfeld von internen Tabellen bis ABAP 7.54«
- Abschnitt 9.9, »Änderungen und Neuerungen im Umfeld von Open SQL bis ABAP 7.54«
- Abschnitt 10.4, »Änderungen und Neuerungen beim Zugriff auf SAP-HANA-Entwicklungsobjekte bis ABAP 7.54«
- Abschnitt 11.15, »Änderungen und Neuerungen in ABAP Objects bis ABAP 7.54«

Zusätzlich sind zwei neue Kapitel zum Buch hinzugekommen, deren Lektüre ich Ihnen empfehle:

- Kapitel 23, »Business Object Processing Framework«
- Kapitel 29, »ABAP-Programmiermodelle«

Interessant ist hier vor allem Kapitel 29, weil es aufzeigt, wie sich SAP die Entwicklung mit ABAP in den nächsten zehn Jahren vorstellt.

Leseplan für Interessenten an den neuen Technologien

Um einen Überblick über SAP HANA und die neue Web-Frontend-Technologie zu erhalten, ist insbesondere Teil VI mit Kapitel 30, »SAP HANA«, und Kapitel 31, »SAPUI5, SAP Fiori und SAP Gateway«, von Bedeutung. In Kapitel 31 wird der Fokus auf die Entwicklung von OData-Services mit ABAP gelegt. Kapitel 10, »Zugriff auf SAP-HANA-Entwicklungsobjekte«, fasst die ABAP-Anweisungen zusammen, die Sie für den Zugriff auf die SAP-HANA-Datenbank benötigen. Vor allem aber mit den in Kapitel 2, »Die ABAP Development Tools«, beschriebenen ADT als Nachfolger der ABAP Workbench sollten Sie sich vertraut machen.

Leseplan für Experten

Sie wissen bereits, welche sprachlichen Neuerungen es bis Release 7.54 gibt, und haben diese Anweisungen auch bereits verinnerlicht? Dann sind Sie hier richtig!

Informieren Sie sich z. B. über den neuen SAP List Viewer mit integriertem Datenzugriff (ALV with Integrated Data Access, IDA, siehe Abschnitt 19.3). Informationen zur Anwendung von Mesh-Typen finden Sie in Abschnitt 8.11, »Meshes«.

Kapitel 24 behandelt Themen der fortgeschrittenen Programmierung, insbesondere Parallelisierung, dynamische Programmierung und dynamisches SQL. Außerdem finden Sie hier Informationen zu persistenten Klassen und Shared Objects sowie der Arbeit mit XML und JSON. Kapitel 28 zeigt Ihnen, wie Sie Entwurfsmuster auf ABAP Objects anwenden können. Gerade die Implementierung des Model-View-Controllers und des Observer-Musters wird Sie hier interessieren. Zusätzlich lohnt sich die Kenntnis des Business Object Processing Frameworks (BOPF), das Sie in Kapitel 23 beschrieben finden.

Ihr individueller Leseweg

Abweichend von den hier vorgestellten Leseplänen können Sie sich dem Buch natürlich auf vielen unterschiedlichen Wegen nähern. Da das Buch eine breite Zielgruppe ansprechen und gerade für den Alltag nützlich sein soll, habe ich versucht, mehrere Zugänge zum Buch zu ermöglichen. Stöbern Sie doch einfach einmal durch das Inhaltsverzeichnis, oder suchen Sie nach den Themen, die Sie interessieren im Index.

Im Anhang finden Sie außerdem ein Glossar mit den wichtigsten Fachbegriffen sowie nützliche Übersichten mit ABAP-Anweisungen, wichtigen Klassen und Funktionsbausteinen, Transaktionscodes, technischen Tabellen und vieles mehr.

Bleibt mir nur noch, Ihnen viel Erfolg und Spaß bei der Arbeit mit dem Buch zu wünschen. Ich hoffe, es hilft Ihnen dabei, die kleinen und großen Herausforderungen des Alltags zu meistern. Die Zusammenstellung des Inhalts basiert vor allem auf meiner Tätigkeit als ABAP-Entwickler und meinen Erfahrungen als Trainer. Mein Dank gilt daher insbesondere denjenigen Teilnehmern meiner Kurse, die immer eine Frage mehr gestellt und mir auf diese Weise neue Sichtweisen eröffnet haben. So gesehen ist das vorliegende Buch ein Gemeinschaftswerk von und für (angehende) ABAPer. Ich freue mich daher auch über jeden Kommentar, der dazu beiträgt, das Buch weiter zu verbessern und abzurunden. Schreiben Sie mir gerne unter der E-Mail-Adresse *abap.felix.roth@gmail.com*.

In diesem Sinne: `COMMIT WORK`!

Felix Roth

Auf einen Blick

Inhalt

TEIL II Der Kern der Sprache ABAP

9 Zugriff auf Datenbanken 287

10 Zugriff auf SAP-HANA-Entwicklungsobjekte 375

11 Die ABAP-Objects-Syntax 389

12 Reports und Selektionsbildschirme 427

TEIL IV Fortgeschrittene Programmiertechniken

Anhang 1065