

ABAP[®]

Das umfassende Handbuch

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

3., aktualisierte Auflage zu ABAP 7.57

Felix Roth

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 fündig!

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 angebotenen, 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)

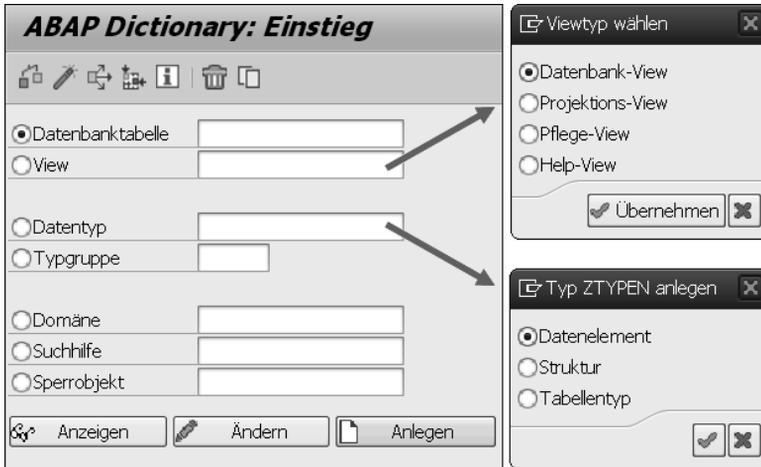


Abbildung 3.1 Einstiegsbildschirm des ABAP Dictionarys

Darüber hinaus bietet das ABAP Dictionary noch etwas verstecktere 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 Ta-

belle 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.

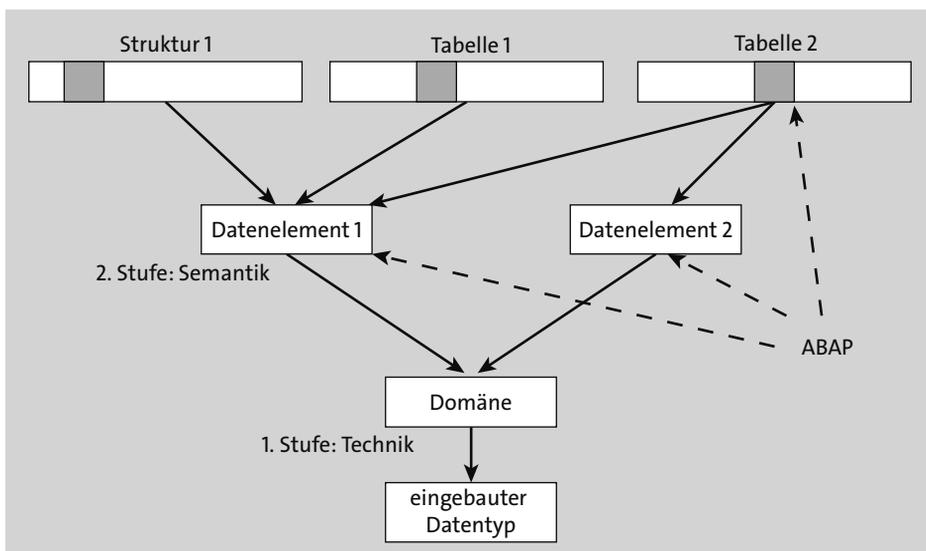


Abbildung 3.2 Das zweistufige Domänenprinzip

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 Kantineangestellte. 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).

Eigenschaften	Definition	Wertebereich
Format		
Datentyp	DBC	Rechen- oder Betragsfeld mit Komma und Vorzeichen
Zahl der Stellen	4	
Dezimalstellen	2	
Ausgabeeigenschaften		
Ausgabelänge	6	
Konvert.-Routine		
<input checked="" type="checkbox"/> Vorzeichen		
<input type="checkbox"/> Kleinbuchstaben		

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

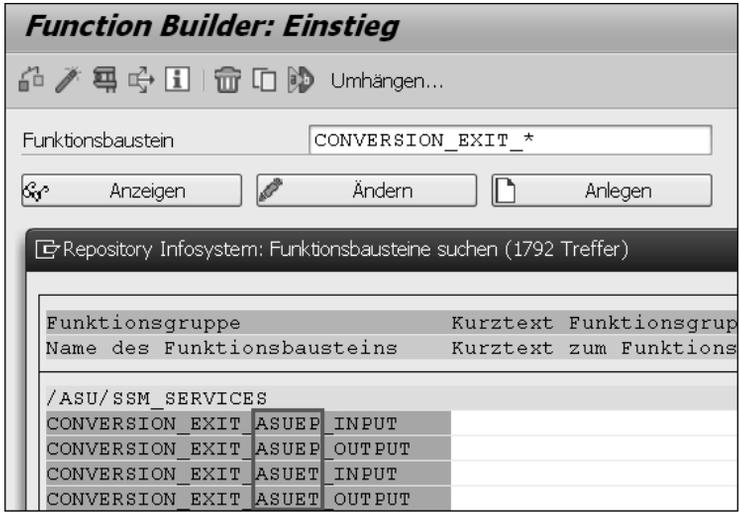


Abbildung 3.4 Konvertierungs-Exits finden

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 (-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.

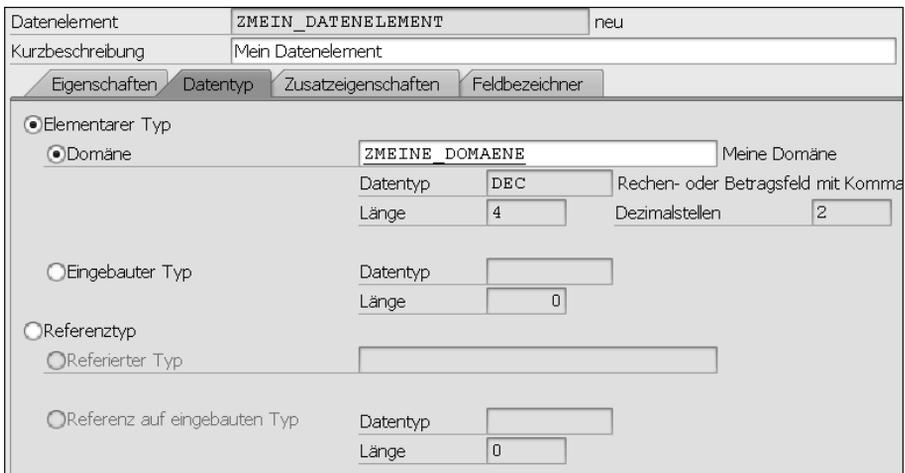


Abbildung 3.6 Datentyp eines Datenelements

Sie können hier aus den folgenden Datentyparten auswählen:

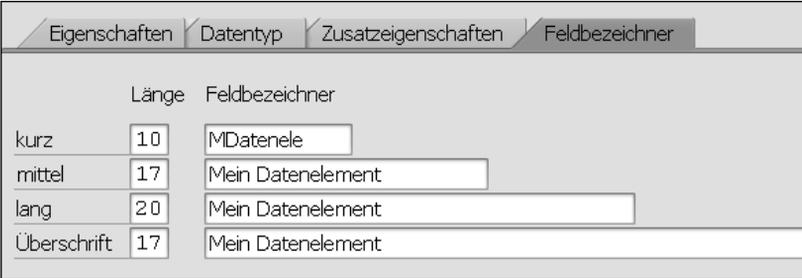
- **Domäne** (siehe Abschnitt 3.1)
- **Eingebauter Typ**
- **Referenztyp** (kann eine Referenz auf eine Klasse, ein Interface oder auf einen eingebauten Typ sein)

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).



	Länge	Feldbezeichner
kurz	10	MDatenele
mittel	17	Mein Datenelement
lang	20	Mein Datenelement
Überschrift	17	Mein Datenelement

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-

Option-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.

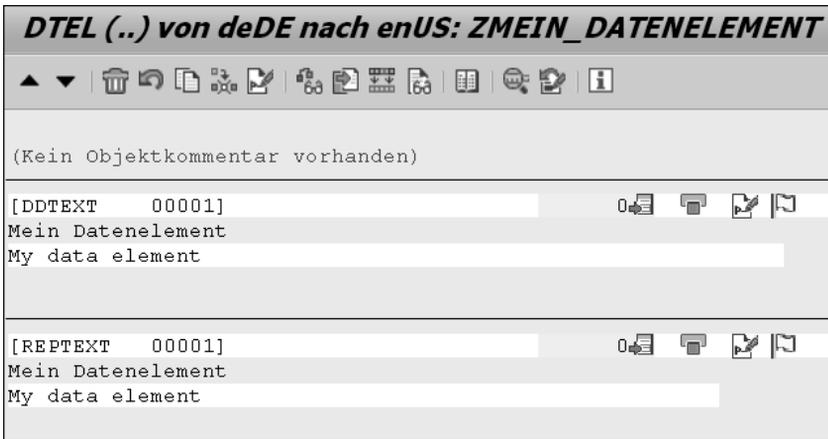


Abbildung 3.9 Übersetzung von Textelementen

4. Sie haben darüber hinaus die Möglichkeit, sich über den Button **Quelltexte in SAP-term 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 7.4.4 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 Referenztable und das Referenzfeld.

Dies schauen wir uns in den folgenden beiden Abschnitten an.

Komponente	Typisierungsart	Komponententyp	Datentyp	Länge	DezSt...	Kurzbeschreibung
MATNR	Type	MATNR	CHAR	18	0	Materialnummer
MEINS	Type	MEINS	UNIT	3	0	Basismengeneinheit
MTART	Type	MTART	CHAR	4	0	Materialart
BRGEW	Type	BRGEW	QUAN	13	3	Bruttogewicht

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«.

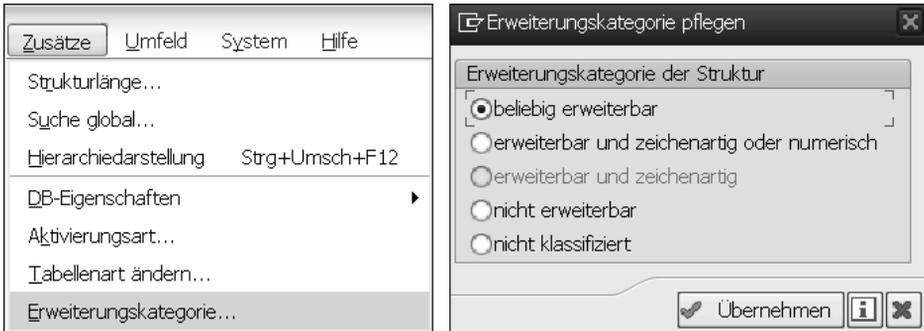


Abbildung 3.11 Erweiterungskategorie einer Struktur

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.

Eigenschaften		Komponenten		Eingabehilfe/-prüfung		Währungs-/Mengenfelder	
Komponente	Typisierung	Datentyp	Referenztable	Referenzfeld	Suchhilfe		1 / 4
MATNR	Type	CHAR					Materialnummer
MEINS	Type	UNIT					Basismengeneinheit
MTART	Type	CHAR					Materialart
BRGEW	Type	QUAN	ZMEINE STRUKTUR	MEINS			Bruttogewicht

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.

Komponente	Typisierungsart	Komponententyp	Datentyp	Länge	De
SIGN	Type	DDSIGN	CHAR	1	
OPTION	Type	DDOPTION	CHAR	2	
LOW	Type	MATNR	CHAR	18	
HIGH	Type	MATNR	CHAR	18	

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)?
- Ausprägungen der Tabellenfelder auf der Registerkarte **Felder:** Welche Spalten soll die Datenbanktabelle haben (siehe Abschnitt 3.5.8)?

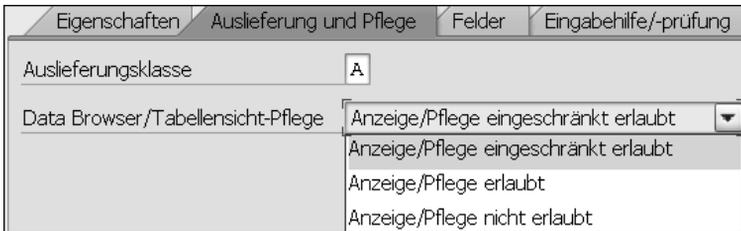


Abbildung 3.17 Registerkarte »Auslieferung und Pflege« im Tabellen-Editor

Darüber hinaus gibt es seit der Einführung von SAP HANA die 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, »Pflagedialoge«, beschrieben, während Sie in Kapitel 17, »Das Transportwesen«, eine Erläuterung des Transportmechanismus finden.

Auslieferungsklasse	Kurzbeschreibung
A	Anwendungstab. (Stamm- und Bewegungsdaten)
C	Customizingtabelle, Pflege nur durch Kunden, kein SAP Import

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, »Pflagedialoge«) 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

Dictionary: Technische Einstellungen pflegen

Überarbeitet<->Aktiv

Name	ZMEINE_TABELLE	Transparente Tabelle
Kurzbeschreibung	Meine Tabelle	
Letzte Änderung	ABAP033	25.06.2016
Status	neu	nicht gesichert
Logische Speicher-Parameter		
Datenart	APPL0	<input type="checkbox"/>
Größenkategorie	0	

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):

- **Einzeleinträge 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.

Feld	Key	Initi...	Datenelement	Datentyp
MANDT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MANDT	CLNT
MATNR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MATNR	CHAR
MTART	<input type="checkbox"/>	<input type="checkbox"/>	MTART	CHAR
MEINS	<input type="checkbox"/>	<input type="checkbox"/>	MEINS	UNIT
ERSDA	<input type="checkbox"/>	<input type="checkbox"/>	ERSDA	DATS
BRGEW	<input type="checkbox"/>	<input type="checkbox"/>	BRGEW	QUAN

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.

Speicherungsart

Column Store Row Store Undefined

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 9.1.1.

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 Auswahlmnü 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.

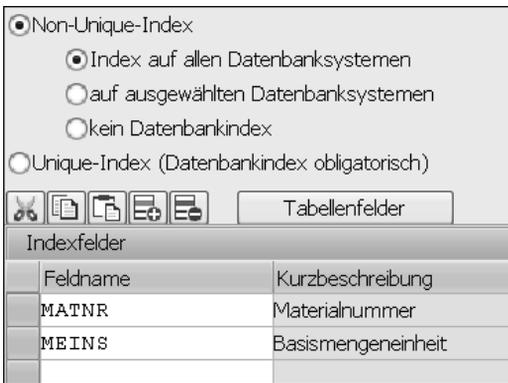


Abbildung 3.24 Definition der Indexfelder

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).

Typgruppe	ZMTYG aktiv
Kurzbeschreibung	Meine Typgruppe
Eigenschaften Quelltext	
1	TYPE-POOL zmtyg.
2	▣ TYPES: BEGIN OF zmtyg_material,
3	matnr TYPE matnr,
4	mtart TYPE mtart,
5	meins TYPE meins,
6	END OF zmtyg_material.

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.

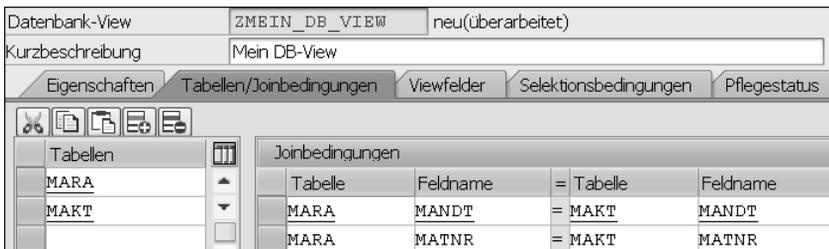


Abbildung 3.26 Join-Bedingung eines Views

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 Feldliste nach der SELECT-Anweisung. Die View-Felder können Sie entweder manuell in die Tabelle eintragen oder sie über den Button **Tabellenfelder** automatisch hinzufügen lassen.

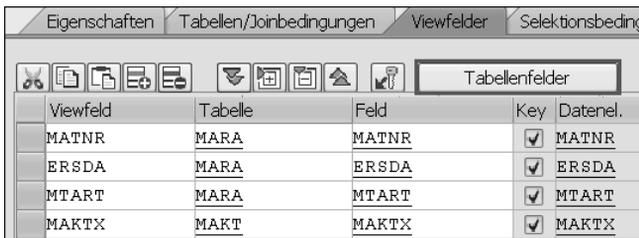


Abbildung 3.27 Felder eines Views

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.

Eigenschaften				Tabellen/Joinbedingungen				Viewfelder				Selektionsbedingungen			
<div style="border: 1px solid gray; padding: 2px;"> ✂ 📄 📄 📄 📄 📄 </div> <div style="border: 1px solid gray; padding: 2px; margin-top: 2px;"> Tabellenfelder </div>															
Tabelle				Feldname				Operator				Vergleichswert			
MAKT				SPRAS				EQ				'D'			

Abbildung 3.28 Selektionsbedingung eines Views

Aktivieren (🔍) Sie gegebenenfalls erneut. Nun können Sie Ihren View testen. Dies ist über den Button **Datenanzeige** (📄) 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«).

Feldname	Key	Datenelement	Mod	DTyp	Länge	Kurzbeschreibung
MATNR	<input checked="" type="checkbox"/>	MATNR	<input type="checkbox"/>	CHAR	18	Materialnummer
ERSDA	<input checked="" type="checkbox"/>	ERSDA	<input type="checkbox"/>	DATS	8	Erstellungsdatum
MTART	<input checked="" type="checkbox"/>	MTART	<input type="checkbox"/>	CHAR	4	Materialart

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«).

The screenshot shows the configuration for a 'Pflege-View' (ZMEIN_PFL_VIEW) in SAP. The 'Pflegestatus' tab is selected. The 'Zugriff' (Access) section has four radio buttons: 'nur lesen', 'lesen, ändern, löschen und einfügen' (selected), 'lesen und ändern', and 'lesen und ändern (zeitabhängige Views)'. The 'Auslieferungsklasse' (Delivery Class) is 'A Anwendungstab. (Stamm- und Bewegungsdaten)'. The 'Data Browser/Tabellensicht-Pflege' (Data Browser/Table Maintenance) dropdown is set to 'Anzeige/Pflege erlaubt'.

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).

Gen. Tabellen-Pflegedialog pflegen: Generierungsumgebung

Bildnummer(n) suchen

Tabelle/View: ZMEINE_TABELLE

Technische Angaben zum Dialog

Berechtigungsgruppe: &NC& ohne Berecht.gruppe

Berechtigungsobjekt: S_TABU_DIS

Funktionsgruppe: ZMEINE_TABELLE

Paket:

Pflegebilder

Pflegetyp: einstufig
 zweistufig

Pflegebildnummer: Übersichtsbild
Einzelbild

Angaben zum Datentransport des Dialogs

Aufzeichnungsroutine: Standard Aufzeichnungsroutine
 keine oder individuelle Aufzeichnungsroutine

Abgleichkennzeichen: automatisch abgleichbar Hinweis

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.

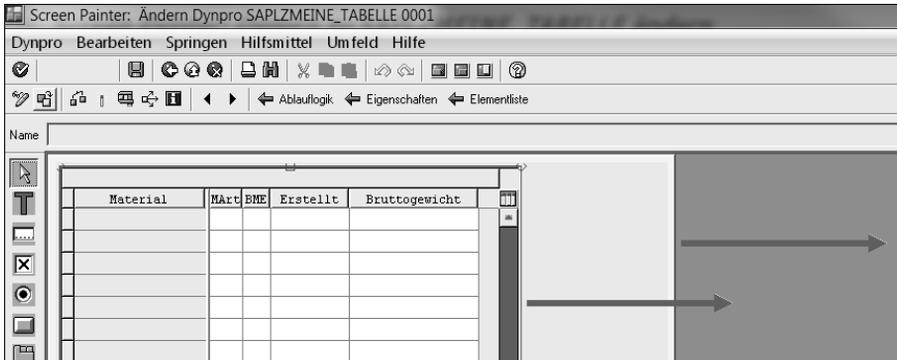


Abbildung 3.33 Verbreiterung eines Pflegedialogs im Screen Painter

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.

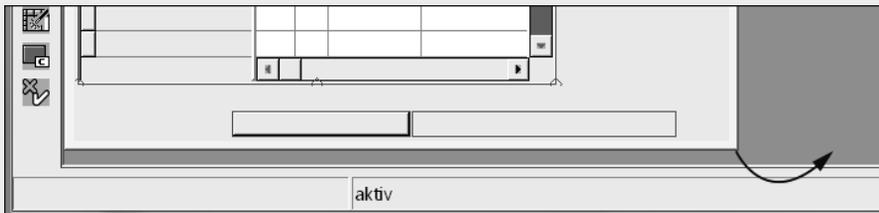


Abbildung 3.34 Dynpros im Screen Painter verbreitern

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.

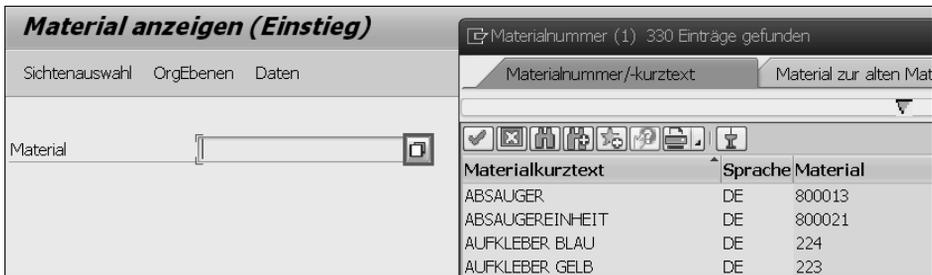


Abbildung 3.35 Eine Suchhilfe für die Materialnummer

Innerhalb des ABAP Dictionary 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 Werteinschränkung angezeigt werden soll oder ob die Ergebnisliste direkt angezeigt werden soll.

Elementare Suchhilfe	ZMEINE_SUCHHILFE	aktiv
Kurzbeschreibung	Meine Suchhilfe	
Eigenschaften		
Definition		
Datenbeschaffung		Dialogverhalten
Selektionsmethode	MARA	Dialogtyp
Texttabelle	MAKT	Dialog abhängig von Wertemenge
		Kurzanwahl
		<input type="checkbox"/>

Abbildung 3.36 Selektionsmethode und Dialogverhalten einer Suchhilfe

- Darüber hinaus müssen Sie die **Suchhilfefparameter** 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.

Parameter									
Suchhilfefparameter	IMP	EXP	LPos	SPos	SAnz	Datenelement	M...	Defaultwert	
MATNR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	1	<input type="checkbox"/>	MATNR	<input type="checkbox"/>		
MAKTX	<input checked="" type="checkbox"/>	<input type="checkbox"/>	2	2	<input type="checkbox"/>	MAKTX	<input type="checkbox"/>		
	<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>		<input type="checkbox"/>		

Abbildung 3.37 Definition von Suchhilfefparametern

- Über die Spalte **IMP** definieren Sie, welche Werte aus Ihrer Datenquelle übernommen werden, also in der Wertetabelle angezeigt werden könnten.
- Über die Spalte **EXP** definieren Sie, welche Werte in die Maske übernommen werden.
- 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 1** und anschließend das Feld MAKTX für den **Materialkurztext 2**, siehe Abbildung 3.38). Soll der Parameter nicht auf der Trefferliste erscheinen, lassen Sie dieses Feld frei.
- 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 3** und dann das Feld MAKTX für den **Materialkurztext 4**, siehe Abbildung 3.38). Soll der Parameter nicht auf der Suchmaske erscheinen, so lassen Sie dieses Feld frei.
- Zum Schluss **Aktivieren**  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 Suchhilfefparameter definieren (d. h. bestimmen, welche Werte übernommen und welche zurückgegeben werden). Diese Suchhilfefparameter tragen Sie auf der Registerkarte **Definition** ein (siehe Abbildung 3.39).

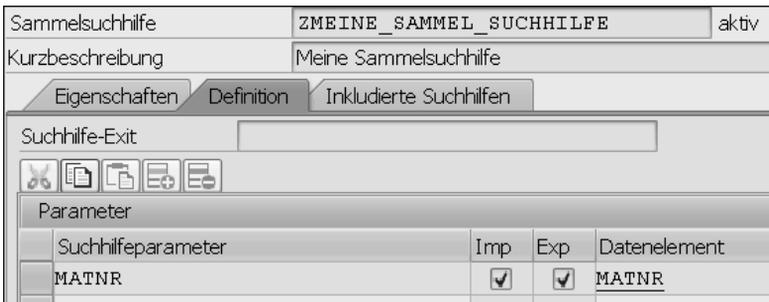


Abbildung 3.39 Suchhilfefparameter einer Sammelsuchhilfe

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 Suchhilfefparameter den Suchhilfefparametern 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**.

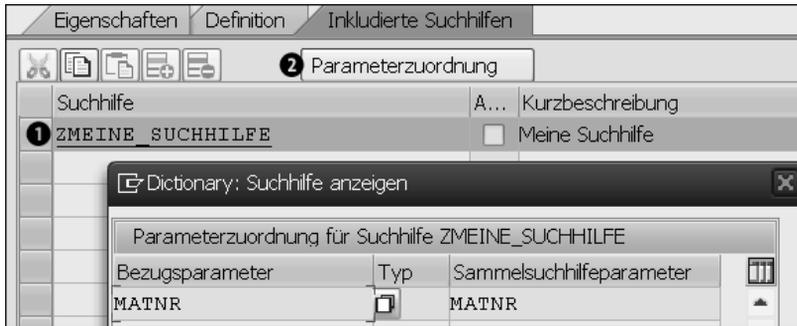


Abbildung 3.40 Parameterzuordnung in einer Sammelsuchhilfe

5. **Aktivieren**  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.

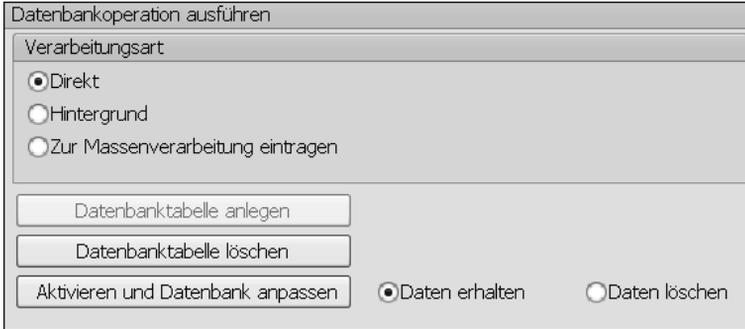


Abbildung 3.41 Datenbankoperationen mit dem Datenbank-Utility-Tool

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.

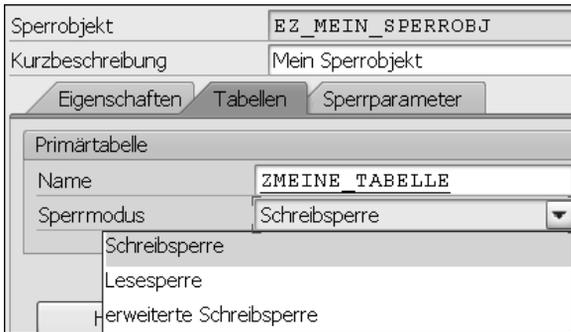


Abbildung 3.42 Sperrmodi eines Sperrobjekts

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

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

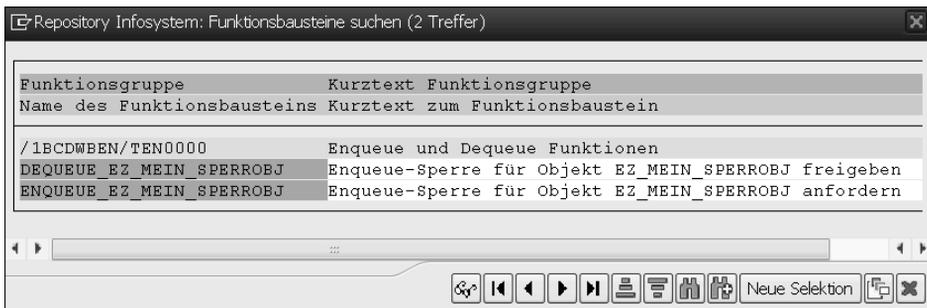


Abbildung 3.43 Sperrbausteine eines Sperrobjekts

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.

Inhalt

Einleitung	33
------------------	----

TEIL I Die Werkzeugkiste des ABAP-Entwicklers

1 Die ABAP Workbench 45

1.1 Die Werkzeuge der ABAP Workbench	46
1.2 Der Object Navigator	47
1.3 Der Repository Browser	50
1.3.1 Übergeordnete Objektliste	51
1.3.2 Favoriten anlegen	51
1.4 Der ABAP Editor	52
1.4.1 Den neuen ABAP Editor aktivieren	52
1.4.2 Einen Report anlegen	53
1.4.3 Die Funktionsleiste	56
1.4.4 Die (Auto-)Vervollständigung	58
1.4.5 Der Pretty Printer	61
1.4.6 Die Musterfunktion	62
1.4.7 Der Package Builder	64
1.4.8 ABAP-Beispiele	66

2 Die ABAP Development Tools 69

2.1 SAPs Eclipse-Strategie	69
2.2 Installation und Konfiguration	71
2.2.1 Installation von Eclipse	72
2.2.2 Konfiguration des Backend-Systems	74
2.3 Ein System anbinden	75
2.4 Einen Report anlegen	76
2.5 Die Menüleiste und wichtige Tastaturkürzel	77

2.6	Der Pretty Printer und weitere Quellcodefunktionen	81
2.7	Die Musterfunktion	81
2.7.1	Muster für Funktionsbausteine	82
2.7.2	Muster für den Aufruf von Methoden	82
2.8	Der Debugger	83
2.8.1	Die Menüleiste des Debuggers	84
2.8.2	Die Variablen- und Outline-Anzeige	85
2.8.3	Breakpoints	86
2.8.4	Watchpoints	87
2.8.5	Interne Tabellen	88
2.9	Dokumentation mit ABAP Doc	89
2.9.1	Beispiel	91
2.9.2	Quick Fix	91
2.9.3	Dokumentation exportieren	92
2.9.4	Verlinkung zu Kurztexten des SAP GUI	92
2.10	Refactoring-Funktionen	93
2.10.1	Umbenennung	93
2.10.2	Eine Methode extrahieren	94
2.10.3	Quick Fix	94
3	Das ABAP Dictionary	97
<hr/>		
3.1	Domänen	99
3.1.1	Das zweistufige Domänenprinzip	99
3.1.2	Domänen anlegen	100
3.1.3	Konvertierungsroutinen	101
3.1.4	Wertebereich einer Domäne	102
3.2	Datenelemente	104
3.2.1	Feldbezeichner	105
3.2.2	Übersetzung	105
3.2.3	Zusatzeigenschaften	107
3.3	Strukturen	107
3.3.1	Erweiterungskategorie	108
3.3.2	Referenztabellen und das Referenzfeld	109
3.4	Tabellentypen	109
3.4.1	Ranges-Tabellentypen anlegen	110

3.5	Datenbanktabellen	112
3.5.1	Datenbanktabellen anzeigen	112
3.5.2	Datenbanktabellen anlegen	112
3.5.3	Auslieferungsklasse	113
3.5.4	Tabellensicht-Pflege	114
3.5.5	Datenart	114
3.5.6	Größenkategorie	115
3.5.7	Pufferung	115
3.5.8	Felder ausprägen	116
3.5.9	Speicherart von Datenbanken mit SAP HANA	117
3.6	Indizes	118
3.7	Typgruppen	119
3.8	Views	120
3.8.1	Datenbank-View	121
3.8.2	Projektions-View	122
3.8.3	Pflege-View	123
3.8.4	Help-View	124
3.9	Pflegedialoge	124
3.9.1	Pflegedialog anlegen	124
3.9.2	Pflegedialog verbreitern	126
3.10	Suchhilfen	128
3.10.1	Elementare Suchhilfe	128
3.10.2	Sammelsuchhilfe	130
3.11	Datenbank-Utility-Tool	131
3.12	Das Sperrkonzept	132
4	Transaktionen	135
4.1	Transaktionen anlegen	136
4.1.1	Dialogtransaktion	137
4.1.2	Reporttransaktion	137
4.1.3	ABAP-Objects-Transaktion	138
4.1.4	Variantentransaktion	138
4.1.5	Parametertransaktion	139
4.2	Transaktionen mit ABAP aufrufen	140
4.2.1	Verwendung von CALL TRANSACTION	140

5 Der Function Builder 143

- 5.1 Der Aufbau eines Funktionsbausteins** 143
 - 5.1.1 Eigenschaften 144
 - 5.1.2 Die Schnittstelle 144
 - 5.1.3 Ausnahmen 146
 - 5.1.4 Der Quellcode 146
- 5.2 Einen Funktionsbaustein anlegen** 146
- 5.3 Funktionsbausteine testen** 147
- 5.4 Funktionsgruppen** 148
 - 5.4.1 Funktionsgruppe anlegen 148
 - 5.4.2 Aufbau einer Funktionsgruppe 149
 - 5.4.3 Lebensdauer einer Funktionsgruppe 150

6 Der Class Builder 151

- 6.1 Klassen anlegen** 151
 - 6.1.1 Vererbung 152
 - 6.1.2 Interfaces 153
 - 6.1.3 Freunde 154
 - 6.1.4 Attribute 155
 - 6.1.5 Methoden 156
 - 6.1.6 Ereignisse 159
 - 6.1.7 Typen 160
 - 6.1.8 Aliases 160
 - 6.1.9 Konstruktoren anlegen 161
 - 6.1.10 Eine Klasse testen 162
 - 6.1.11 Klassen direkt bearbeiten 162
- 6.2 Ausnahmeklassen anlegen** 163
- 6.3 Interfaces anlegen** 164

TEIL II Der Kern der Sprache ABAP

7 Die ABAP-Grundbefehle	167
7.1 Syntaxregeln	168
7.2 Kommentare	169
7.2.1 Mehrere Zeilen kommentieren	170
7.3 Die SAP-Hilfe	170
7.4 Datendeklaration	171
7.4.1 Felder	172
7.4.2 Konstanten	175
7.4.3 Konstanten mit Inline-Deklaration	175
7.4.4 Strukturen	176
7.4.5 Zuweisung mit MOVE-CORRESPONDING	177
7.4.6 Aufzählungstypen	178
7.4.7 Feldsymbole	180
7.4.8 Unterschied zwischen TYPE und LIKE	182
7.5 Inline-Deklarationen	183
7.6 Typdefinitionen	185
7.6.1 Felder	185
7.6.2 Strukturen	186
7.7 Initialisierung	188
7.7.1 Felder initialisieren	188
7.7.2 Speicherbereich freigeben	188
7.8 Steueranweisungen	189
7.8.1 Die IF-Abfrage	189
7.8.2 Logische Ausdrücke	190
7.8.3 Die CASE-Anweisung	191
7.8.4 Die Anweisung CASE TYPE OF	192
7.8.5 Die DO-Schleife	193
7.8.6 Die WHILE-Schleife	194
7.8.7 Die CHECK-Anweisung	194
7.8.8 Die EXIT-Anweisung	195
7.8.9 Die CONTINUE-Anweisung	195
7.9 Rechenoperationen	196
7.9.1 Mathematische Funktionen	197
7.9.2 Berechnungszuweisungen	198

7.10	Ausgabeeweisungen	199
7.10.1	Die Anweisung WRITE	200
7.10.2	Das Muster für die Listenausgabe	200
7.11	Meldungen	201
7.12	Mit Zeichenketten arbeiten	203
7.12.1	Vergleich von Zeichenketten	203
7.12.2	Verkettungsoperatoren	204
7.12.3	Teilfeldzugriff	206
7.12.4	Teilzeichenketten finden	207
7.12.5	Teilzeichenketten ersetzen	209
7.12.6	Eingebaute Funktionen für Zeichenketten	211
7.12.7	Zeichenketten-Templates	212
7.13	Konstruktorausdrücke	216
7.13.1	VALUE: Erzeugung von Werten	217
7.13.2	REF: Referenzen besorgen	219
7.13.3	EXACT: Verlustfreie Zuweisung/Berechnung	220
7.13.4	CONV: Konvertierung von Werten	220
7.13.5	COND: Bedingte Ausdrücke	221
7.13.6	SWITCH: Bedingte Ausdrücke	222
7.13.7	NEW: Instanziierung	223
7.13.8	CORRESPONDING: Mapping von Strukturen und internen Tabellen	224
7.13.9	Der Zusatz LET	227
7.13.10	Der Zusatz BASE	227
7.14	Operandenpositionen	227
7.14.1	Funktionen und Ausdrücke für Lesepositionen	228
7.14.2	Ausdrücke für Schreibpositionen	229
7.15	Änderungen und Neuerungen bis ABAP 7.57	229
8	Mit internen Tabellen arbeiten	231
<hr/>		
8.1	Tabellenarten	232
8.1.1	Standardtabellen	233
8.1.2	Sortierte Tabellen	233
8.1.3	Hash-Tabellen	234
8.2	Interne Tabellen definieren	234
8.2.1	Schlüssel definieren	235

8.2.2	Obsolet: Deklaration einer internen Tabelle mit Kopfzeile	236
8.2.3	Ranges-Tabellen definieren	237
8.3	Interne Tabellen initialisieren	238
8.4	Zeilen hinzufügen	239
8.4.1	Daten mit SELECT hinzufügen	239
8.4.2	Zeilen mit APPEND anhängen	239
8.4.3	Zeilen mit INSERT hinzufügen	241
8.4.4	Werte mit VALUE hinzufügen	242
8.4.5	Der Zusatz FOR	243
8.4.6	Gruppierungen mit FOR	245
8.4.7	Der Zusatz LINES OF	249
8.4.8	Hinzufügen mit NEW	250
8.5	Inhalt auslesen	250
8.5.1	Tabellen mit READ TABLE auslesen	251
8.5.2	Tabellenausdrücke	253
8.5.3	Tabellen mit LOOP AT auslesen	257
8.5.4	Gruppieren mit dem Zusatz GROUP BY	259
8.6	Einträge löschen	264
8.7	Inhalt ändern	265
8.7.1	Tabelle mit READ TABLE ändern	265
8.7.2	Tabelle mit Tabellenausdrücken ändern	266
8.7.3	Tabelle mit MODIFY ändern	266
8.7.4	Tabelle mit CORRESPONDING anreichern	268
8.8	Interne Tabellen kopieren	270
8.8.1	Strukturgleiche interne Tabellen kopieren	271
8.8.2	Strukturfremde interne Tabellen kopieren	271
8.9	Interne Tabellen aufbereiten	272
8.9.1	Sortieren mit SORT	272
8.9.2	Virtuelles Sortieren	273
8.9.3	Zusammenfassung mit COLLECT	276
8.9.4	Reduzierungen mit REDUCE	276
8.9.5	Filterungen mit FILTER	278
8.10	Eingebaute Funktionen für interne Tabellen	279
8.11	Änderungen und Neuerungen im Umfeld von internen Tabellen bis ABAP 7.57	281

9	Zugriff auf Datenbanken	285
9.1	Die fünf goldenen Regeln	286
9.1.1	Die goldenen Regeln für SAP HANA	287
9.2	Die Open-SQL-Anweisung SELECT	288
9.2.1	Einträge lesen	288
9.2.2	Gelesene Spalten einschränken	291
9.2.3	Zielspalten angeben	292
9.2.4	Die WHERE-Klausel	293
9.2.5	Ranges-Tabellen und Selektionsoptionen	295
9.2.6	FOR ALL ENTRIES IN: Einschränkung durch interne Tabellen	296
9.2.7	Gruppierung und Sortierung der Ergebnisse	297
9.2.8	Die FIELDS-Klausel	299
9.2.9	Host-Variablen und -ausdrücke	300
9.2.10	Inline-Deklaration	302
9.2.11	Begrenzung der Ergebnismenge mit OFFSET	303
9.2.12	JOIN: Verknüpfung	304
9.2.13	WITH: Allgemeine Tabellenausdrücke	310
9.2.14	UNION: Vereinigung	311
9.2.15	INTERSECT: Schnittmenge	312
9.2.16	EXCEPT: Ausschließen	313
9.2.17	Unterabfragen	313
9.2.18	SELECT auf interne Tabellen	314
9.3	Open-SQL-Ausdrücke	315
9.3.1	CASE-Anweisungen	315
9.3.2	Verknüpfungen von Zeichenketten mit &&	317
9.3.3	Arithmetische Ausdrücke	317
9.3.4	Typumwandlungen mit CAST	318
9.3.5	Elementare Werte	319
9.3.6	Typisierte Literale	320
9.4	Open-SQL-Funktionen	321
9.4.1	Aggregatfunktionen	321
9.4.2	Zeichenkettenfunktionen	324
9.4.3	Numerische Funktionen	326
9.4.4	Datums- und Zeitfunktionen	327
9.4.5	Coalesce-Funktion: Nullwerte ersetzen	330
9.4.6	UUID-Funktion: Erzeugung eindeutiger Schlüssel	331
9.4.7	Umwandlung von Währungen und Einheiten	331
9.5	Ändernde Open-SQL-Anweisungen	334
9.5.1	DELETE: Löschen von Einträgen	334

9.5.2	INSERT: Einträge einfügen	337
9.5.3	UPDATE: Einträge ändern	338
9.5.4	MODIFY: Einfügen oder Ändern	341
9.6	Sekundäre Datenbankverbindungen	343
9.7	Natives SQL	343
9.7.1	EXEC-SQL	344
9.7.2	ABAP Database Connectivity	345
9.8	ABAP Core Data Services (CDS)	346
9.8.1	Anlage eines CDS Views	348
9.8.2	CDS Views mit Parametern	354
9.8.3	CDS-Sprachelemente	357
9.8.4	CDS-Zugriffskontrollen	364
9.8.5	CDS-Annotationen	366
9.8.6	CDS-Assoziationen	369
9.8.7	CDS-Tabellenfunktionen	371
9.8.8	CDS-Hierarchien	374
9.9	Änderungen und Neuerungen im Umfeld von Open SQL bis ABAP 7.57 ...	378
9.10	Änderungen und Neuerungen im Umfeld von CDS bis ABAP 7.57	381
10	Zugriff auf SAP-HANA-Entwicklungsobjekte	385
10.1	Aufruf von SAP-HANA-Views	385
10.1.1	Aufruf mit nativem SQL	386
10.1.2	Aufruf über externe Views	386
10.1.3	Externe Views anlegen	386
10.2	Aufruf von Datenbankprozeduren	388
10.2.1	Aufruf mit nativem SQL	388
10.2.2	Aufruf mit einem Datenbankprozedur-Proxy	390
10.2.3	Datenbankprozedur-Proxy anlegen	390
10.3	ABAP Managed Database Procedures (AMDP)	391
10.3.1	ABAP Managed Database Procedures anlegen	392
10.3.2	Datenbankfunktionen anlegen	394
10.3.3	BAAls für ABAP Managed Database Procedures	395
10.4	Änderungen und Neuerungen beim Zugriff auf SAP-HANA-Entwicklungsobjekte bis ABAP 7.57	396

11 Die ABAP-Objects-Syntax	399
11.1 Grundaufbau einer Klasse	400
11.1.1 Instanziierung von Klassen	401
11.1.2 Instanziierung mit CREATE OBJECT	401
11.1.3 Instanziierung mit NEW	402
11.2 Sichtbarkeiten	403
11.2.1 Komponentensichtbarkeit	403
11.3 Datentypen und Attribute	404
11.3.1 Zugriff auf Attribute	405
11.4 Methoden	405
11.4.1 Methoden implementieren	407
11.4.2 Methoden aufrufen	408
11.5 Konstruktoren	412
11.5.1 Instanzkonstruktor	412
11.5.2 Statischer Konstruktor	413
11.6 Ereignisse	415
11.6.1 Definition von Ereignissen	415
11.6.2 Ereignisse auslösen	415
11.6.3 Definition eines Ereignisbehandlers	416
11.6.4 Ereignisbehandlung registrieren	416
11.6.5 Beispiel für die Definition, das Auslösen und die Behandlung eines Ereignisses	416
11.7 Vererbung	418
11.7.1 Redefinition von Methoden	419
11.8 Klassenarten	420
11.8.1 Abstrakte und finale Klassen	420
11.8.2 Statische Klassen	421
11.8.3 Ausnahmeklassen	421
11.9 Ausnahmen für Methoden	422
11.9.1 Klassenbasierte Ausnahmen	423
11.9.2 Lokale Ausnahmen	426
11.10 Freunde	428
11.11 Interfaces	428
11.11.1 Implementierung eines Interface	429
11.11.2 Verwendung von Interfaces	431
11.12 Das ABAP-Objects-Muster	432

11.13 Casting	433
11.13.1 Casting mit dem Zuweisungsoperator	433
11.13.2 Casting mit dem Casting-Operator	434
11.13.3 Casting mit der Anweisung CAST	434
11.14 Objekttyp überprüfen	434
11.14.1 Die Anweisung IS INSTANCE OF	434
11.14.2 Die Anweisung CASE TYPE OF	435
11.15 Änderungen und Neuerungen in ABAP Objects bis ABAP 7.57	436

12 Reports und Selektionsbildschirme 437

12.1 Ereignisse eines Reports	438
12.2 Eingabeelemente	439
12.2.1 Parameter	440
12.2.2 Checkboxes	441
12.2.3 Radiobuttons	442
12.2.4 Dropdown-Liste	442
12.2.5 Selektionsoptionen	444
12.2.6 Buttons	445
12.2.7 Buttons auf der Funktionsleiste	447
12.2.8 Der Zusatz USER-COMMAND	448
12.3 Strukturierungselemente für den Selektionsbildschirm	449
12.3.1 Blöcke	449
12.3.2 Leerzeilen	450
12.3.3 Horizontale Linien	450
12.3.4 Textausgaben	450
12.3.5 Tabstrips	451
12.3.6 Modifikationsgruppen	452
12.4 Ereignisse eines Selektionsbildschirms	453
12.4.1 Selektionselemente dynamisch ausblenden	455
12.5 Textelemente	457
12.5.1 Zugriff auf Textelemente	457
12.5.2 Textsymbole	458
12.5.3 Selektionstexte	459
12.5.4 Listenüberschriften	460
12.6 Nachrichtenklassen	461
12.6.1 Nachrichtenklasse anlegen	461

12.6.2	Nachricht aufrufen	462
12.6.3	Parametrisierte Nachrichten	463
12.7	Einen Report mit ABAP aufrufen	464
12.7.1	Selektionsparameter übergeben	464
12.7.2	Selektionsparameter frei angeben	465
12.7.3	Übergabe einer Selektionstabelle	465
12.8	SPA-/GPA-Parameter	466
12.8.1	SPA-/GPA-Parameter anlegen und setzen	467
12.8.2	SPA-/GPA-Parameter auslesen	468

13 Strukturierungselemente in ABAP 469

13.1	Unterprogramme	470
13.1.1	Unterprogramm definieren	471
13.1.2	Sichtbarkeitsbereiche von Datendeklarationen	471
13.1.3	Aufruf eines Unterprogramms	472
13.1.4	Parameterübergabe	473
13.2	Makros	476
13.2.1	Makros definieren	477
13.2.2	Makros aufrufen	477
13.3	Includes	478
13.3.1	Include einbinden	478
13.3.2	Top-Include anlegen	478
13.3.3	Include anlegen	480
13.4	Funktionsbausteine	481
13.4.1	Arten von Funktionsbausteinen	481
13.4.2	Aufruf von Funktionsbausteinen	482
13.4.3	Behandlung von lokalen Ausnahmen	484
13.4.4	Behandlung von Ausnahmeklassen	485
13.4.5	Funktionsbausteine finden	485
13.5	Datenkonsistenz	487

14 Die Dynpro-Programmierung 489

14.1	Dynpros anlegen	490
14.1.1	Dynpro gestalten	491

14.1.2	Dynpro aufrufen	493
14.1.3	Zugriff auf Dynpro-Elemente	494
14.2	Ablauflogik eines Dynpros	494
14.2.1	Process Before Output (PBO)	495
14.2.2	PBO-/PAI-Module anlegen	496
14.2.3	GUI-Status	497
14.2.4	GUI-Titel	500
14.2.5	Process After Input (PAI)	501
14.3	SAP Control Framework	502
14.3.1	Custom Control anlegen	505
14.4	Pop-up-Fenster	506
14.4.1	Entscheidungen	506
14.4.2	Textanzeige	507
14.4.3	Werteabfrage	508

TEIL III Techniken zur Qualitätssicherung

15 Tests und Qualitätskontrolle 513

15.1	Der ABAP Debugger	513
15.1.1	Den neuen ABAP Debugger aktivieren	514
15.1.2	Den Debugger starten und beenden	514
15.1.3	Die Oberfläche	517
15.1.4	Die Werkzeuge	518
15.1.5	Steuerung des Debuggers	519
15.1.6	Schnellanzeige der Variablen	521
15.1.7	Vergleichstool	522
15.1.8	Aufrufstack	523
15.1.9	Pop-up-Fenster debuggen	524
15.1.10	Interne Tabellen	525
15.1.11	Debugger-Breakpoints	527
15.1.12	Watchpoints	530
15.1.13	Ausnahmen	531
15.2	Das Debugging-Skript	532
15.2.1	Einen Trigger für das Skript definieren	533
15.2.2	Ein Skript schreiben	534
15.2.3	Das Skript starten und beenden	536
15.2.4	Beispiel: Watchpoints für Feldsymbole	537

- 15.3 Der Code Inspector** 538
 - 15.3.1 Ad-hoc-Prüfung über Transaktion SE80 539
 - 15.3.2 Prüfvariante 539
 - 15.3.3 Objektmenge 541
 - 15.3.4 Inspektion 541
 - 15.3.5 Ergebnisliste 541

- 15.4 ABAP Unit** 542
 - 15.4.1 Grundsätzlicher Aufbau einer Testklasse 543
 - 15.4.2 Systemeinstellungen 546
 - 15.4.3 Assertions 547
 - 15.4.4 Assistent für die Testklassengenerierung 548
 - 15.4.5 Ausführen eines ABAP-Unit-Tests 550
 - 15.4.6 Die Ergebnisanzeige 550
 - 15.4.7 Der ABAP Unit Browser 551

- 15.5 Das ABAP Test Cockpit** 551
 - 15.5.1 Ausführung eines ATC-Tests 552
 - 15.5.2 Die Transaktion ATC 553
 - 15.5.3 Der ATC-Ergebnis-Browser 553

16 Werkzeuge und Tipps zur Performanceanalyse 555

- 16.1 Richtlinien für die ABAP-Entwicklung** 556

- 16.2 Transaktion SAT: Laufzeitanalyse** 558
 - 16.2.1 Laufzeitmessung durchführen 559
 - 16.2.2 Laufzeitmessung auswerten 560
 - 16.2.3 Anzeige der Messungen 561

- 16.3 Transaktion SE30: Die alte Laufzeitanalyse** 562
 - 16.3.1 Performance-Trace im Debugger 563

- 16.4 SQL-Monitor** 564
 - 16.4.1 Transaktion SQLM: Den SQL-Monitor administrieren 565
 - 16.4.2 Transaktion SQLMD: Analyse der Daten 567

- 16.5 SQL Performance Tuning Worklist** 568

- 16.6 Transaktion ST05** 569
 - 16.6.1 SQL-Trace 570
 - 16.6.2 Analyse einer SQL-Anweisung 572

- 16.7 Laufzeitanalyse mithilfe der ABAP-Programmierung** 573

16.7.1	Zeitmessung	573
16.7.2	Fortschrittsanzeige implementieren	574
16.8	Application Log	575
16.8.1	Transaktion SLG0: Ein Application Log anlegen	576
16.8.2	Log mit Nachrichten befüllen	576
16.8.3	Log als Pop-up-Fenster darstellen	579
17	Das Transportwesen	581
17.1	Die SAP-Systemlandschaft	582
17.2	Transportaufträge	585
17.2.1	Transportauftrag anlegen	585
17.2.2	Transportauftrag freigeben und importieren	586
17.2.3	Aufgabe anlegen	590
17.2.4	Zuordnung des Transportauftrags ändern	590
17.2.5	Transportaufträge und Aufgaben löschen	592
17.2.6	Objekte in einen Transportauftrag aufnehmen	592
17.2.7	Transportaufträge verschmelzen	593
17.2.8	Transportauftrag oder Aufgabe finden	594
17.2.9	Freigabe eines Transportauftrags zurücknehmen	594
18	Die Jobverwaltung	597
18.1	Transaktion SM36: Jobs definieren	597
18.1.1	Allgemeine Angaben	598
18.1.2	Startbedingung	599
18.1.3	Schritt (Step) definieren	602
18.2	Transaktion SM37: Jobs überwachen und freigeben	604
18.2.1	Jobs freigeben	604
18.2.2	Freigabe zurücknehmen	604
18.2.3	Spool- und Job-Log anzeigen	605
18.2.4	Einplanung eines Jobs wiederholen	605
18.3	Ereignisse für Jobs	605
18.3.1	Ereignis definieren	605
18.3.2	Ereignis auslösen	606
18.4	Jobs mit ABAP definieren	606

TEIL IV Fortgeschrittene Programmier Techniken

19 Tabellenanzeige mit dem SAP List Viewer (ALV)	611
19.1 Die alte ALV-Anzeige	613
19.1.1 Aufbau des Grundgerüsts	614
19.1.2 Eingabefähigkeit	616
19.1.3 Funktionen	619
19.1.4 Ereignisse	621
19.1.5 Spalten bearbeiten	622
19.1.6 Zellentypen	624
19.1.7 Farbige Hervorhebung	627
19.1.8 Icons	628
19.1.9 ALV-Tabellen sortieren und gruppieren	630
19.1.10 Aggregation	631
19.1.11 Layout speichern	631
19.2 Die neue ALV-Anzeige	632
19.2.1 Aufbau des Grundgerüsts	632
19.2.2 Funktionen	634
19.2.3 Ereignisse	636
19.2.4 Spalten bearbeiten	637
19.2.5 Zellentypen	640
19.2.6 Farbige Hervorhebung	641
19.2.7 Icons	643
19.2.8 ALV-Tabellen sortieren und gruppieren	643
19.2.9 Aggregation	644
19.2.10 Layout speichern	645
19.2.11 Filter	646
19.3 SAP List Viewer mit integriertem Datenzugriff (IDA)	647
19.3.1 Aufbau des Grundgerüsts	648
19.3.2 Funktionen	649
19.3.3 Ereignisse	650
19.3.4 Spalten bearbeiten	651
19.3.5 Zellentypen	656
19.3.6 Icons	657
19.3.7 ALV-Tabelle sortieren und gruppieren	657
19.3.8 Aggregation	657
19.3.9 Layout	658

19.3.10 Filter	659
19.3.11 Textsuche	661
19.4 Mehrere ALV-Tabellen auf einer Oberfläche	662

20 SAP-Schnittstellen 665

20.1 RFC-Funktionsbausteine	666
20.1.1 Funktionsbaustein remote mit ABAP aufrufen	667
20.2 Business-Objekte und BAPIs	668
20.2.1 Business Object Repository und BAPI Explorer	672
20.2.2 BAPIs	673
20.3 Flat Files	679
20.3.1 Dateien schreiben	680
20.3.2 Dateien einlesen	682
20.3.3 Weitere nützliche Funktionen	684
20.4 Webservices (SOAP)	686
20.4.1 WSDL-Dokument	688
20.4.2 Webservices anlegen und finden	688
20.4.3 Webservice konsumieren	696
20.5 Batch Input	702
20.5.1 Aufzeichnung der Transaktion	703
20.5.2 Direkte Ausführung	705
20.5.3 Mappe erstellen	708
20.6 Einführung in die Legacy System Migration Workbench (LSMW)	709

21 SAP-Erweiterungen 711

21.1 User Exits	711
21.2 Customer Exits	714
21.3 Klassische Business Add-ins (BADIs)	718
21.3.1 BADIs mithilfe eines Breakpoints finden	719
21.4 Enhancement Framework	722
21.4.1 Architektur	724
21.4.2 Explizite Erweiterungspunkte	726
21.4.3 Implizite Erweiterungspunkte	729

21.4.4	Klassenerweiterungen	730
21.4.5	Funktionsbaustein-Erweiterungen	732
21.4.6	Erweiterungssektionen	733
21.4.7	Strukturerweiterungen	735
21.4.8	Suchhilfenerweiterungen	736
21.4.9	Indexerweiterungen	737
21.4.10	Einzelwerterweiterungen	738
21.4.11	Debugging von Erweiterungen	739
21.4.12	Transaktion SPAU_ENH: Abgleich von Erweiterungen im Rahmen von Updates	740
21.5	Neue Business Add-ins (BADIs)	742
21.5.1	Aufruf	743
21.5.2	Definition aufrufen	743
21.5.3	Implementierung anlegen	743
21.5.4	Filterwerte	744
21.5.5	Menü-Exit	745
21.5.6	Dynpro-Exit	746
21.6	Switch Framework	755
21.6.1	Architektur	756
21.6.2	Schaltbare Objekte	757
21.7	Suche nach Erweiterungen	757
22	SAP-Formularentwicklung	759
22.1	Der Druckdialog	761
22.2	SAPscript	762
22.2.1	Formular erstellen	765
22.2.2	Druckprogramm erstellen	769
22.3	SAP Smart Forms	770
22.3.1	Formular erstellen	772
22.3.2	Druckprogramm erstellen	778
22.4	SAP Interactive Forms by Adobe	780
22.4.1	Formular erstellen	780
22.4.2	Druckprogramm erstellen	784
22.5	Generierung von PDFs	785
22.5.1	Generierung der internen Tabelle in SAPscript	786
22.5.2	Generierung der internen Tabelle SAP Smart Forms	786

22.5.3	Generierung der internen Tabelle in SAP Interactive Forms by Adobe	787
22.5.4	OTF in PDF konvertieren	789

23 Business Object Processing Framework 791

23.1	Aufbau von Geschäftsobjekten	794
23.1.1	Knoten	794
23.1.2	Knotenelemente	796
23.1.3	Alternative Schlüssel	798
23.1.4	Aktionen	800
23.1.5	Assoziationen	801
23.1.6	Ermittlungen	801
23.1.7	Validierungen	802
23.1.8	Abfragen	803
23.1.9	Berechtigungsprüfungen	805
23.2	Anwendung der Consumer-API	805
23.2.1	Lesen von Knoteninstanzen	806
23.2.2	Konvertieren von alternativen Schlüsseln	810
23.2.3	Modifizieren von Objekten	812

24 Fortgeschrittene Programmier Techniken 821

24.1	Object Services	822
24.1.1	Persistente Klasse anlegen	822
24.1.2	Datenbanktabelle lesen	826
24.1.3	Datenbanktabelle aktualisieren	827
24.1.4	Query anlegen	828
24.1.5	Neuen Eintrag in der Datenbanktabelle anlegen	829
24.1.6	Löschen eines neuen Eintrags	830
24.2	Mit XML und JSON arbeiten	830
24.2.1	Exkurs: XML und JSON	830
24.2.2	Konvertierung von ABAP in JSON/XML	831
24.2.3	Konvertierung von JSON/XML in ABAP	832
24.2.4	Erzeugung eines XML-Dokuments	833
24.2.5	Objekte serialisieren	834

24.3	Daten im Memory ablegen	835
24.3.1	Die Anweisungen EXPORT und IMPORT	836
24.3.2	Shared Objects	838
24.4	Parallelisierung	842
24.5	Dynamische Erzeugung von Datenobjekten	844
24.5.1	Anonymes Datenobjekt mit CREATE DATA erzeugen	844
24.5.2	Anonymes Datenobjekt mit NEW erzeugen	845
24.5.3	Beispiel: Dynamische ALV-Tabelle erzeugen	846
24.6	Runtime Type Services (RTTS)	850
24.6.1	Strukturen: Klasse CL_ABAP_STRUCTDESCR	852
24.6.2	Interne Tabellen: Klasse CL_ABAP_TABLEDESCR	853
24.6.3	Referenzdatentypen: Klasse CL_ABAP_REFDESCR	854
24.6.4	Klassen: Klasse CL_ABAP_CLASSDESCR	854
24.6.5	Interfaces: Klasse CL_ABAP_INTFDESCR	855
24.6.6	Elementare Datentypen: Klasse CL_ABAP_ELEMDESCR	856
24.6.7	Beispiel: Eine interne Tabelle nach Microsoft Excel exportieren	856
24.7	Dynamisches SQL	860
24.7.1	Dynamische Selektionsliste	860
24.7.2	Dynamische FROM-Klausel	861
24.7.3	Dynamische WHERE-Klausel	862
24.8	Das ABAP Daemon Framework (ADF)	862
24.8.1	Anlegen eines Daemons	863
24.8.2	Starten eines Daemons	864
24.8.3	Übergabe von Parametern	867
24.8.4	Dem Daemon eine Nachricht senden	868
24.8.5	Den Daemon stoppen	869
24.9	ABAP Channels	870
24.9.1	ABAP Messaging Channels	870
24.9.2	ABAP Push Channels	876

TEIL V Objektorientierte Programmierung

25	Grundlagen der Objektorientierung	885
25.1	Einführung für ABAP-Entwickler	885
25.2	Klassen und Objekte	891
25.2.1	Statische Klassen	893

25.3	Instanziierung	894
25.3.1	Konstruktoren	896
25.4	Datenkapselung	897
25.4.1	Freunde	899
25.5	Ereignisse	900
25.6	Vererbung	902
25.6.1	Redefinition	904
25.6.2	Klassenhierarchien	905
25.7	Klassenarten	907
25.7.1	Abstrakte Klassen	908
25.7.2	Finale Klassen	908
25.8	Interfaces	908
25.9	Polymorphie	912
25.10	Zusammenfassung	914

26 Unified Modeling Language (UML) 917

26.1	Anwendungsfalldiagramm	918
26.1.1	Akteure	919
26.1.2	Anwendungsfälle	920
26.1.3	Beziehungen	920
26.1.4	Beispieldiagramm	922
26.1.5	Textuelle Beschreibung	923
26.2	Klassendiagramm	923
26.2.1	Attribute	926
26.2.2	Operationen	926
26.2.3	Beziehungen zwischen Klassen	927
26.2.4	Vom Anwendungsfall zum Klassendiagramm	929
26.2.5	Beispieldiagramm	930

27 Anwendungsentwicklung – wo fange ich an? 933

27.1	Anforderungsermittlung	935
27.1.1	Ermittlung	936
27.1.2	Spezifikation	937

27.1.3 Verhaltensmodellierung	939
27.1.4 Validierung	940
27.2 Analyse	940
27.2.1 Die Klassenmodellierung	941
27.2.2 Verhaltensmodellierung	943
27.2.3 Verifikation	944
27.3 Entwurf	944

28 Entwurfsmuster 947

28.1 Singleton	948
28.2 Fabrikmethode	950
28.2.1 Factory-Methode beim SAP List Viewer	953
28.3 Model View Controller	955
28.3.1 Model-Klasse	958
28.3.2 View-Klasse	959
28.3.3 Controller-Klasse	960
28.3.4 Hauptprogramm	961
28.3.5 Ersetzung des Views	961
28.4 Fassade	963
28.5 Observer	964
28.5.1 Haupttabelle	967
28.5.2 Abstrakte Detailtabelle	968
28.5.3 Konkrete Detailtabelle	968
28.5.4 Hauptprogramm	969
28.6 Objektorientierte Reports	971

TEIL VI Ein Blick über den Tellerrand: Was Sie als ABAP-Entwickler sonst noch kennen sollten

29 ABAP-Programmiermodelle 977

29.1 Das klassische Programmiermodell	980
29.2 Die Innovation für die Zukunft	982

29.3	Das ABAP-Programmiermodell für SAP Fiori	983
29.4	Das ABAP RESTful Application Programming Model	987

30 SAP HANA 999

30.1	Überblick	1000
30.2	Architektur	1001
30.2.1	Die In-Memory-Technologie	1002
30.2.2	Spaltenorientierte Speicherung	1002
30.2.3	Wertekomprimierung	1004
30.3	Migration auf SAP HANA	1005
30.4	SAP-HANA-Objekte	1006
30.4.1	Attribute View	1007
30.4.2	Analytic View	1008
30.4.3	Calculation View	1009
30.4.4	Stored Procedures	1010
30.5	Volltextsuchen	1010
30.5.1	Volltextindex anlegen	1011
30.5.2	Fuzzy-Suche	1013
30.5.3	Linguistische Suche	1014
30.5.4	Eingabefelder für die Vorschlagssuche	1015

31 SAPUI5, SAP Fiori und SAP Gateway 1019

31.1	SAPUI5	1020
31.2	SAP Fiori	1022
31.2.1	SAP Fiori Launchpad	1022
31.3	OData	1024
31.3.1	Beispiel	1025
31.3.2	Metadatendokument	1026
31.3.3	Aufbau eines OData-Service	1027
31.3.4	Abfrageoptionen	1028
31.4	SAP Gateway	1030
31.4.1	Embedded Deployment	1030

31.4.2	Zentrales Deployment ohne Entwicklung	1031
31.4.3	Zentrales Deployment mit Entwicklung	1031
31.5	Entwicklung eines OData-Service	1032
31.5.1	Entwicklung	1033
31.5.2	Veröffentlichung	1040
31.5.3	Testen	1042
31.5.4	Fehleranalyse	1044
31.6	Überblick über die Implementierung der CRUDQ-Methoden	1044
31.6.1	Auslesen der Schlüsselfelder	1044
31.6.2	Auslesen des HTTP-Request-Bodys	1045
31.6.3	Abfrageoptionen \$skip und \$top	1046
31.6.4	Abfrageoption \$count	1047
31.6.5	Abfrageoption \$inline-count	1047
31.6.6	Abfrageoption \$filter	1048
31.6.7	Abfrageoption \$select	1049
31.6.8	Abfrageoption \$orderby	1049
31.6.9	Meldungen ausgeben	1050

32 Andere SAP-Webtechnologien 1053

32.1	Business Server Pages (BSP)	1054
32.1.1	BSP-Anwendung mit HTML anlegen	1056
32.1.2	BSP-Anwendung mit HTMLB anlegen	1059
32.2	Web Dynpro ABAP	1061
32.2.1	Web-Dynpro-Component anlegen	1063
32.2.2	Ausgabetable definieren	1065
32.2.3	Methode zur Datenselektion implementieren	1068
32.2.4	Context mit View verbinden	1072
32.2.5	Ergebnistabelle anlegen	1075
32.2.6	Logik für den Button implementieren	1076
32.2.7	Web-Dynpro-Anwendung anlegen	1078
32.3	Internet Communication Framework	1079

Anhang	1083
A Das SAP-Flugdatenmodell	1085
B Übersicht der ABAP-Anweisungen	1087
C Eingebaute Datentypen	1097
D Transaktionscodes	1099
E Wichtige Systemfelder	1103
F Technische Tabellen	1105
G Nützliche Funktionsbausteine	1107
H Klassen	1111
I Namenskonventionen für die Programmierung	1113
J Systemglossar und Suche nach fremdsprachigen SAP-Begriffen	1117
K Glossar	1119
Der Autor	1125
Index	1127

Alle ABAP-Werkzeuge und -Techniken

Programm- und Dialogentwicklung

Sie lernen die ABAP-Grundbefehle, den Umgang mit internen Tabellen, den Zugriff auf Datenbanken und die Syntax von ABAP – alles aktuell zu ABAP 7.57. So beherrschen Sie den Umgang mit Reports, Selektionsbildschirmen, Klassen und Funktionsbausteinen bald mühelos.

Hilfreiche Werkzeuge

Felix Roth stellt Ihnen nützliche Techniken und Tools für die ABAP-Entwicklung vor: Debugger, Performanceanalyse-Tools, Transportwesen, SAP List Viewer (ALV), Core Data Services (CDS) und vieles mehr.

Objektorientierte Programmierung

Die objektorientierte Programmierung mit ABAP wird Ihnen mit diesem Buch keine Schwierigkeiten mehr bereiten. Verstehen Sie, welche Ideen dahinterstecken, und erlernen Sie den sicheren Umgang mit Klassen und Methoden.

Auf einen Blick

- ABAP Workbench und ABAP Development Tools (Eclipse)
- ABAP Dictionary
- Datenbankzugriffe und neue ABAP-SQL-Funktionen
- CDS und BOPF
- Reports und Selektionsbildschirme
- Tests und Performanceanalyse
- Klassen, Methoden, Ereignisse und Interfaces
- ABAP RESTful Application Programming Model
- Mit Glossar und nützlichen Übersichten

»Eine ausgezeichnete Ressource, ich habe schon lange nach einem solchen Buch gesucht!«

Leser-Feedback zur Voraufgabe

Der Autor

Felix Roth ist selbstständiger ABAP-Trainer, -Entwickler und Berater (LOOP AT Consulting). Er berät Kunden in verschiedenen Entwicklungsprojekten und beschäftigt sich vor allem mit den neuesten SAP-Technologien. Seit 2014 hält er regelmäßig SAP-Schulungen, unter anderem beim Rheinwerk Verlag.

