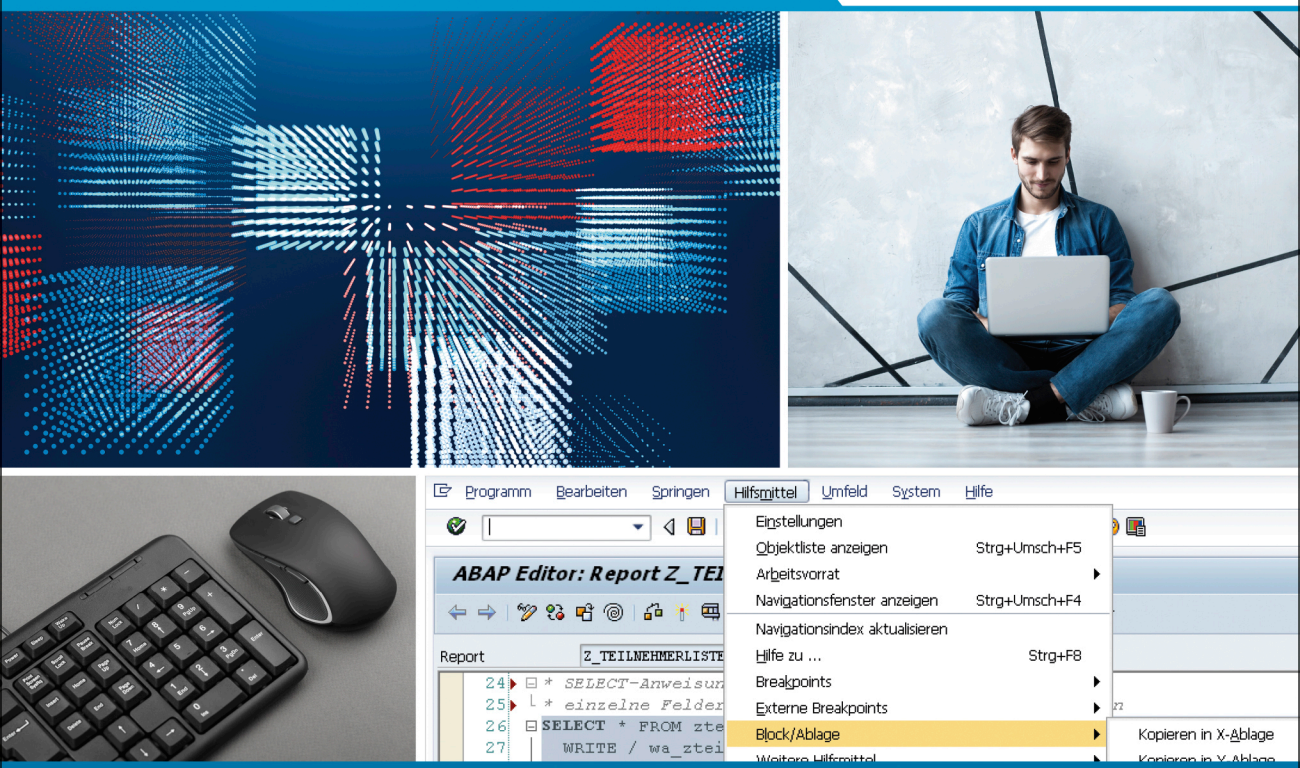
SAP PRESS



Einstieg in ABAP®

6. Auflage

- Ihre Einführung für einen schnellen Lernerfolg
- Kommentierte Codebeispiele und anschauliche Screenshots
- Inklusive Debugging, ABAP in Eclipse und CDS Views

Mit Programmierbeispielen zum Download

Thorsten Franz
Karl-Heinz Kühnhauser

Rheinwerk Publishing

Kapitel 2
ABAP Dictionary

Für die Beispielanwendung dieses Buchs benötigen wir Daten, mit denen wir arbeiten können. Dazu erstellen wir als Erstes eine Tabelle im ABAP Dictionary. Dabei lernen Sie ganz nebenbei, wie Sie mit dem ABAP Dictionary umgehen, um Objekte anzulegen.

Bereits in der Standardauslieferung des SAP-Systems sind Tausende Tabellen enthalten. Dennoch benötigen Kunden zusätzlich eigene Tabellen für individuelle Anforderungen. Die wichtigsten Arbeiten im Zusammenhang mit dem Anlegen und Pflegen von Tabellen erledigen wir daher gleich am Anfang.

In diesem Kapitel legen wir eine Tabelle an und füllen diese. Auf diese Tabelle werden Sie in den weiteren Kapiteln immer wieder zugreifen. Sie ist die grundlegende Tabelle unserer Beispielanwendung, einer Verwaltungsanwendung für die Teilnehmer von IT-Kursen. In der Tabelle legen Sie für jeden Teilnehmer einen Datensatz an, ändern und pflegen ihn. Schrittweise erarbeiten Sie sich so das Wissen für immer schwierigere Aufgaben.

Tabellen in der Praxis

Natürlich werden in der Praxis die Daten der SAP-Komponenten nicht in einer einzigen Tabelle gepflegt, sondern auf viele Tabellen verteilt, die über Beziehungen zu komplexen Datenmodellen verknüpft sind. Doch um die Grundprinzipien zu verstehen, genügt uns vorerst eine Tabelle.

2.1 Einstieg in das ABAP Dictionary

Rufen Sie zunächst das ABAP Dictionary auf – entweder über den Navigationspfad **SAP Menü • Werkzeuge • ABAP Workbench • Entwicklung • Dictionary** oder über den Transaktionscode SE11. Der Einstiegsbildschirm des ABAP Dictionary wird nun angezeigt (siehe Abbildung 2.1).

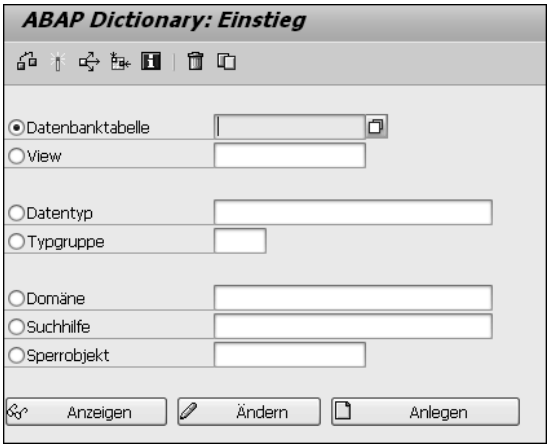


Abbildung 2.1 Einstieg in das ABAP Dictionary

ABAP Dictionary außerhalb des SAP GUI

Mit den *ABAP Development Tools* (ADT), auch *ABAP in Eclipse* genannt, besteht seit einigen Jahren die Möglichkeit, innerhalb der Open-Source-Entwicklungsumgebung *Eclipse* ABAP-Programme zu entwickeln. Eclipse ist eine kostenlose, Java-basierte Entwicklungsumgebung, die vor allem bei Java-Entwicklern sehr beliebt ist.

Für den Einstieg sollten Sie die klassische Entwicklungsumgebung über das SAP GUI verwenden, die daher auch die Grundlage für die Beispiele in diesem Buch darstellt. Im Zusammenhang mit dem ABAP Dictionary ist jedoch zu erwähnen, dass es seit SAP NetWeaver Application Server (AS) ABAP 7.40 einige Objekttypen gibt (insbesondere CDS Views), die nur in Eclipse gepflegt werden können. Diesen widmen wir uns in Kapitel 15, »Core Data Services zur Abbildung von Datenmodellen«.

2.1.1 Datenbanktabellen

Vom Einstiegsbild aus können Sie sich ein Dictionary-Objekt – das heißt eine Datenbanktabelle, einen View, einen Datentyp, eine Typgruppe, eine Domäne, eine Suchhilfe oder ein Sperrobjekt – anzeigen lassen, es ändern oder anlegen. Vorerst soll Sie hier nur die Arbeit mit Datenbanktabellen interessieren.

In SAP-Systemen gibt es drei Arten von Datenbanktabellen:

- transparente Tabellen
- Pool-Tabellen
- Cluster-Tabellen

Bei Pool- und Cluster-Tabellen liegen die Daten in einem zugeordneten Tabellen-Pool bzw. -Cluster auf der physischen Datenbank. Beide Tabellentypen dienen der Ablage spezieller Daten, wie beispielsweise Programmparameter, Dynpro-Folgen oder Dokumentationen, und können Daten aus mehreren Tabellen gemeinsam in einem Pool bzw. Cluster halten. Pool- und Cluster-Tabellen dienen üblicherweise nicht der Speicherung betriebswirtschaftlicher Anwendungsdaten, da diese grundsätzlich in transparenten Tabellen liegen. Darüber hinaus sind Pool- und Cluster-Tabellen seit der Einführung von SAP HANA weitgehend obsolet und bestehen mehr als im Einzelfall schwer abzulösende historische Altlasten denn als reguläre Tabellentypen fort.

Transparente Tabellen

Wie beschrieben, arbeiten wir ausschließlich mit SAP-Werkzeugen, um auf der physischen Datenbank Objekte anzulegen oder zu pflegen. Transparente Tabellen sind die einzige Tabellenart, bei der 1:1 zu der im ABAP Dictionary hinterlegten Tabellendefinition auch entsprechende physische Tabellendefinitionen auf der physischen Datenbank angelegt werden.

Ein Tabellen-Pool besteht auf der physischen Datenbank aus nur einer einzigen Tabelle. Alle Tabellen, die diesem Pool zugeordnet sind, heißen *Pool-Tabellen*. In der Pool-Tabelle sind alle Sätze der Tabellen des Pools abgelegt, die über den Namen der Pool-Tabelle und Schlüsselfelder identifiziert werden.

Ein Tabellen-Cluster besteht aus mehreren *Cluster-Tabellen*. Das Tabellen-Cluster fasst Datensätze mit gleichem Schlüssel aus mehreren Cluster-Tabellen zu einem physischen Datensatz zusammen. Die Anzahl der Datensätze wird hierdurch geringer; gleichzeitig wird ein Datensatz jedoch länger, sodass häufig Fortsetzungssätze angelegt werden müssen.

Nur mit Open SQL

Eine 1:1-Umsetzung wie bei transparenten Tabellen erfolgt bei Cluster- und Pool-Tabellen nicht; deshalb müssen auch keine technischen Einstellungen oder Indizes gepflegt werden. Für die Verarbeitung dieser Tabellentypen sind aber die im ABAP Dictionary gespeicherten Informationen zwingend notwendig. Im »Brandfall« (z. B. wenn in einem System fehlerhafte Daten korrigiert werden müssen, aber der Transport eines ABAP-Programms zur Durchführung der Datenkorrektur aus irgendwelchen Gründen nicht möglich ist) können diese Tabellentypen deshalb auch nicht über Native SQL, sondern immer nur mit Open SQL verarbeitet werden. Dabei gelten im Vergleich zu transparenten Tabellen erhebliche Einschränkungen, weil große Teile des SQL-Befehlssatzes wie etwa Tabel-

len-Joins, Aggregatfunktionen und DISTINCT-Klauseln nicht zur Verfügung stehen und die Tabellen auch nicht in CDS Views verwendet werden können.

2.1.2 Tabelle anlegen und pflegen

Beim Anlegen einer neuen Tabelle müssen Sie sich zuerst einen Namen für diese Tabelle überlegen. Der Tabellenname hat maximal 16 Stellen, wobei nicht zwischen Groß- und Kleinschreibung unterschieden wird. Beachten müssen Sie, dass – wie bei fast allen Objekten – unterschiedliche *Namensräume* für Standard-SAP-Objekte und kundenindividuelle Objekte verwendet werden.

Nach den Namenskonventionen beginnen Kundenobjekte mit einem Z, Y oder einem sogenannten *Präfixnamensraum*. Ein Präfixnamensraum ist eine von Schrägstrichen eingerahmte, eindeutig bei SAP reservierte Buchstabenfolge, die dem Namen der Entwicklungsobjekte vorangestellt wird und die kein anderer SAP-Kunde in seinen kundenindividuellen Objekten verwenden darf – beispielsweise /SAPPRESS/. Damit wird garantiert, dass Objektnamen wirklich eindeutig sind und es beispielsweise bei Unternehmensfusionen nicht zu Namenskonflikten kommt. Für unsere Übungszwecke sind jedoch die herkömmlichen Kundennamensräume Z und Y ausreichend. Deshalb nennen Sie Ihre erste Datenbanktabelle »ZTEILNEHMER« und klicken auf den Button **Anlegen** (siehe Abbildung 2.2).

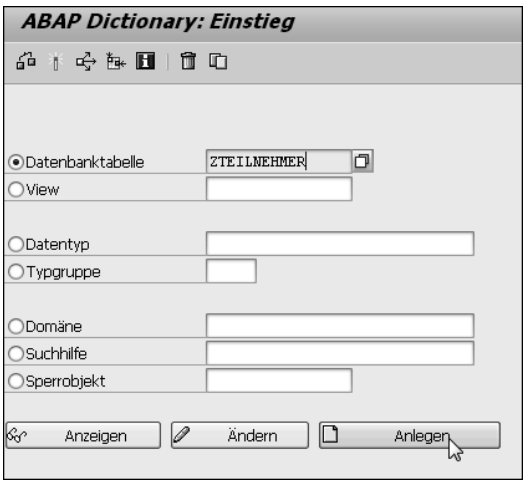


Abbildung 2.2 Datenbanktabelle anlegen

Der Pflegebildschirm des ABAP Dictionary mit fünf Registerkarten erscheint: **Eigenschaften**, **Auslieferung und Pflege**, **Felder**, **Eingabehilfe/-prüfung** sowie **Währungs-/**

Mengenfelder. Im Feld **Kurzbeschreibung** in den Kopfdaten geben Sie einen aussagekräftigen, erklärenden Text ein, z. B. »Tabelle der Kursteilnehmer«. Die Registerkarte **Eigenschaften** enthält die Paketzuordnung bzw. die Kennzeichnung als lokales Objekt (Paket `$TMP`) und wird beim ersten Speichern des Objekts automatisch befüllt.

Auf der Registerkarte **Auslieferung und Pflege** müssen zwei Felder ausgefüllt werden:

- **Auslieferungsklasse**
  Die **Auslieferungsklasse** regelt den Transport der Datensätze der Tabelle bei der Installation oder einem Upgrade, bei einer Mandantenkopie oder dem Transport zu einem anderen SAP-System. Je nach Auslieferungsklasse haben SAP und Kunden unterschiedliche Schreibrechte.

  Geben Sie in diesem Feld ein »A« für eine Anwendungstabelle der Stamm- und Bewegungsdaten ein. *Stammdaten* sind Anwendungsdaten, die sich relativ selten ändern, wie beispielsweise Materialnummern oder Personalnummern. *Bewegungsdaten* ändern sich häufig, wie beispielsweise Rechnungspositionen.

  Die Tabellen der verschiedenen Auslieferungsklassen werden bei Mandantenkopien, Neuinstallationen oder Upgrades unterschiedlich behandelt. Bei einer Mandantenkopie werden beispielsweise Stamm- und Bewegungsdaten grundsätzlich nicht mit kopiert.

- **Data Browser/Tabellensicht-Pflege**
  Die Einstellung im Feld **Data Browser/Tabellensicht-Pflege** regelt, in welchem Umfang die SAP-Standardpflegewerkzeuge für die Anzeige und Pflege der Datensätze verwendet werden dürfen. Wenn Sie diese Rechte einschränken, kann es geschehen, dass beispielsweise im ABAP Dictionary keine neuen Datensätze für Ihre Tabelle erfasst werden können.

  Aus diesem Grund wählen Sie für dieses Feld die Option **Anzeige/Pflege erlaubt**, da Sie so für Ihre Arbeit im ABAP Dictionary alle nötigen Rechte haben und neue Datensätze erfassen oder Datensätze verändern und löschen können (siehe Abbildung 2.3).

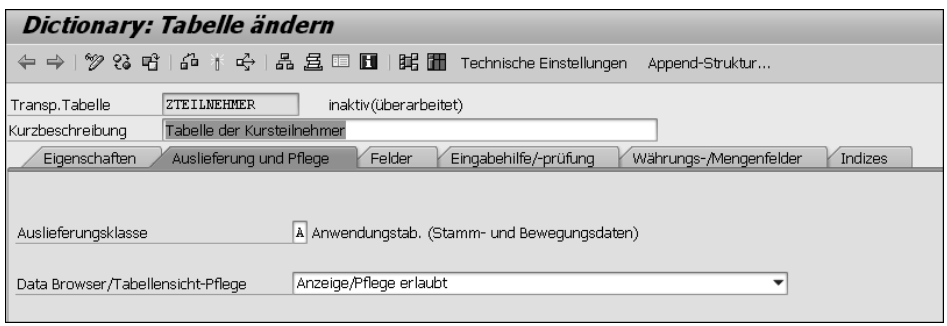


Abbildung 2.3 Einstellungen zur Auslieferung und Pflege der Tabelle

Bevor Sie damit jedoch beginnen, sollten Sie den bisherigen Stand der Arbeit sichern, am einfachsten über den Button **Sichern** (💾) oder – wie der Button-Tooltip schon sagt – über die Tastenkombination Strg + S.

Im folgenden Dialogfenster treffen Sie eine wichtige Entscheidung: Sie legen fest, ob Ihre Tabelle in ein anderes SAP-System transportiert werden kann oder nicht. Soll die Tabelle transportiert werden, müssen Sie ein *Paket* angeben. Ein Paket bündelt zusammengehörige Objekte der ABAP Workbench und definiert die Transportschicht, die wiederum bestimmt, ob Objekte einem transportierbaren Änderungsauftrag zu einem Zielsystem zugeordnet werden oder ob sie auf dem aktuellen lokalen SAP-System verbleiben.

[+]

Transport – ja oder nein?

Es gilt, gut zu überlegen: Im »wirklichen Leben«, das heißt in einer mehrstufigen Systemlandschaft, würden Sie von den Systemverantwortlichen den Namen des Pakets erfragen und eintragen, weil Sie z. B. auf dem Test- und Entwicklungssystem Objekte anlegen, die später für ein Produktivsystem bestimmt sind. Für Ihre ersten Gehversuche im Übungsbeispiel ist dies jedoch nicht notwendig, sondern es genügt, wenn die Tabelle und Ihre ersten Reports lokal auf dem Test- und Entwicklungssystem verbleiben.

Speichern Sie die Tabelle als **Lokales Objekt** (siehe Abbildung 2.4). Sie wird damit automatisch dem Paket $TMP zugeordnet. Objekte dieses Pakets werden nie transportiert und unterliegen auch keiner Versionsverwaltung.

Abbildung 2.4 Tabelle als lokales Objekt speichern

Im nächsten Schritt tragen Sie auf der Registerkarte **Felder** (siehe Abbildung 2.5) die Felder der Tabelle ein und legen fest, ob ein Feld ein Schlüsselfeld ist, ob es auf der Datenbank mit einem Initialwert versehen werden soll und welches Datenelement die betriebswirtschaftlichen Attribute des Felds beschreibt:

- **Feld**
  Bei der Wahl des Feldnamens müssen Sie keine Namenskonventionen beachten; dennoch sollte der Feldname möglichst sinnvoll gewählt sein. Da ein Feld der Tabelle immer über die Tabelle angesprochen wird, und zwar in der Form `Tabellenname-Feldname`, ist der Unterschied zwischen Kundenfeldern und SAP-Feldern hier kein Problem. Der Feldname hat maximal 30 Stellen.

Abbildung 2.5 Tabellenfelder pflegen

- **Key**
  Unter **Key** legen Sie fest, ob ein Feld ein *Schlüsselfeld* ist. Schlüsselfelder müssen lückenlos zusammenhängend am Anfang der Tabelle stehen. Über Schlüsselfelder wird eine Zeile eindeutig identifiziert, und sie dienen als Sortier- und Suchkriterium. Maximal sind 16 Schlüsselfelder möglich.
- **Initialwerte**
  Der Eintrag **Initialwert** ist interessant, falls das Feld in einem Datensatz leer ist, das heißt keinen gültigen Wert enthält. Soll das Feld auf der Datenbank, je nach Typ, mit dem entsprechenden Initialwert vorbelegt werden, beispielsweise eine Integerzahl mit dem Wert 0, muss hier ein Eintrag erfolgen; im Übrigen bleibt das Feld wirklich leer. Der Feldinhalt kann aber auch jederzeit nachträglich mit einem gültigen Wert gefüllt werden.

  Die Vorbelegung mit Initialwerten ist in der Regel vorzuziehen, da in dem Feld auf der Datenbank sonst Nullwerte auftreten können. Diese sind in der Programmierung leicht mit initialen Werten zu verwechseln, was zu unangenehmen Fehlern führen kann, weil beispielsweise Datenbankabfragen und Joins nicht wie erwartet funktionieren. Sie sollten daher in der Regel ein Häkchen in der Spalte **Initialwert** setzen.

- **Datenelement**
  In der Spalte **Datenelement** tragen Sie den Namen eines Datenelements ein bzw. wählen ein vorgegebenes Datenelement aus. Im Datenelement sind einige Attribute eines Felds wie Länge und Typ hinterlegt.

Betriebswirtschaftliche und technische Attribute

Eine Besonderheit des ABAP Dictionary ist, dass die Eigenschaften von Feldern, die *Feldattribute*, in der Regel nicht direkt beim Feld abgelegt werden, sondern getrennt davon in eigenen Dictionary-Objekten. Man unterscheidet hierbei betriebswirtschaftliche und technische Attribute:

- Die betriebswirtschaftlichen Attribute eines Felds liegen im zugehörigen Datenelement.
- Die technischen Attribute eines Felds liegen in der zugehörigen Domäne.

Den Namen des Datenelements tragen Sie deshalb in der Spalte **Datenelement** der Tabellenpflege ein, den der Domäne beim Datenelement. Letzteren Vorgang betrachten wir in Abschnitt 2.2.1, »Datenelement anlegen«, noch näher.

Diese Zerlegung hat unter anderem den Vorteil, dass Datenelemente und Domänen als eigene Dictionary-Objekte mehrfach verwendbar sind: Felder verschiedener Tabellen können auf dasselbe Datenelement verweisen, und Änderungen von Feldattributen können an einer Stelle zentral gepflegt werden. Abhängige Objekte werden dann zur Laufzeit automatisch nachgeneriert.

2.2 Datenelemente und Domänen

Wir durchlaufen nun exemplarisch die Reihenfolge Feld – Datenelement – Domäne und legen dabei alle Objekte an. Als Beispiel dient zunächst ein Feld, in dem Sie den Namen des Systemmandanten speichern, das heißt des Mandanten, mit dem Sie sich angemeldet haben. Objekte, wie z. B. Tabellen, sind mandantenunabhängig zentral in der Datenbank abgelegt und könnten auch aus anderen Mandanten heraus angesprochen werden. Für die Identifizierung des richtigen Datensatzes ist die Einschränkung auf den richtigen Mandanten eine große Hilfe.

2.2.1 Datenelement anlegen

Das erste Feld unserer Tabelle `ZTEILNEHMER` soll `MANDANT` heißen und ein Schlüsselfeld sein. Als Datenelement verwenden Sie das bestehende Element `MANDT`. Tragen Sie diese Anga-

ben ein, und klicken Sie auf den Button **Sichern** (Strg + S). Beachten Sie anschließend die rechten, blau hinterlegten Spalten der ersten Zeile (siehe Abbildung 2.6). Da das Datenelement MANDT bereits aktiv existiert, wurden alle Attribute des Datenelements und der entsprechenden Domäne automatisch übernommen, in unserem Beispiel der **Datentyp**, die **Länge**, die Anzahl der Dezimalstellen (**DezStellen**) und eine **Kurzbeschreibung**. Mehr Arbeit haben Sie für dieses Tabellenfeld nicht zu leisten.

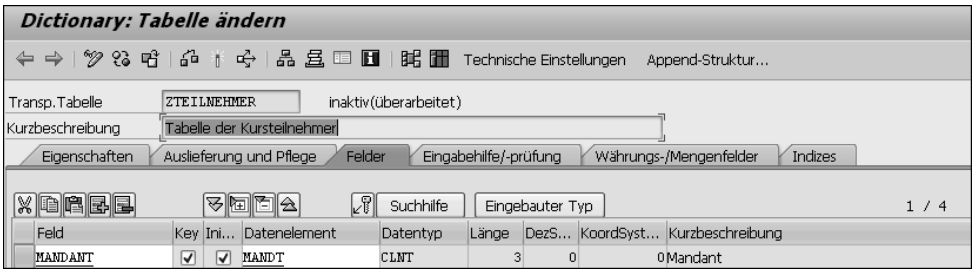


Abbildung 2.6 Tabellenfeld mit bestehendem Datenelement anlegen

Ein bisschen umfangreicher wird die Angelegenheit, wenn sowohl das Datenelement als auch die Domäne neu gefunden werden sollen. Denken Sie aber bei der weiteren Arbeit daran, dass Sie das Rad nicht immer wieder neu erfinden müssen und es durchaus eine sinnvolle und bessere Alternative sein kann, wie gezeigt auf bestehende Objekte wie Datenelemente oder Domänen zurückzugreifen.

Das zweite Feld der Tabelle soll eine eindeutige Teilnehmernummer zwischen 1 und 99.999 beinhalten. Als Feldnamen tragen Sie »TNUMMER« ein, für den Namen des Datenelements müssen Sie nur den Kundennamensraum beachten, im Übrigen können Sie ihn frei wählen – beispielsweise ZTNUMMER. Nachdem Sie beides eingetragen und auch dieses Feld über ein Häkchen in der Spalte **Key** zum Schlüsselfeld gemacht haben, sichern Sie Ihre Arbeit wieder.

Im Gegensatz zur ersten Zeile bleiben dieses Mal die blau hinterlegten Felder hinter der Spalte **Datenelement** unausgefüllt, da das Datenelement ja noch gar nicht existiert. Um beim Anlegen eines Datenelements nicht umständlich durch die ABAP Workbench navigieren zu müssen, verwenden Sie an dieser Stelle die sogenannte *Vorwärtsnavigation*. Hierzu klicken Sie doppelt auf das Objekt, das Sie interessiert – in unserem Fall das Datenelement ZTNUMMER –, und das System fragt, ob Sie das Datenelement anlegen möchten (siehe Abbildung 2.7).

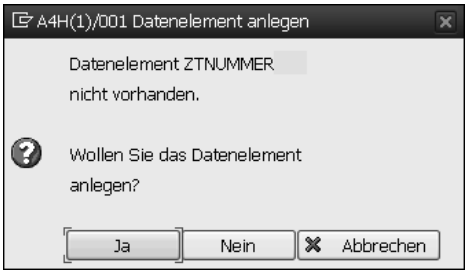


Abbildung 2.7 Objekt anlegen

Diese Frage bestätigen Sie mit **Ja**, und Sie landen im Dialog zur Pflege des Datenelements (siehe Abbildung 2.8).

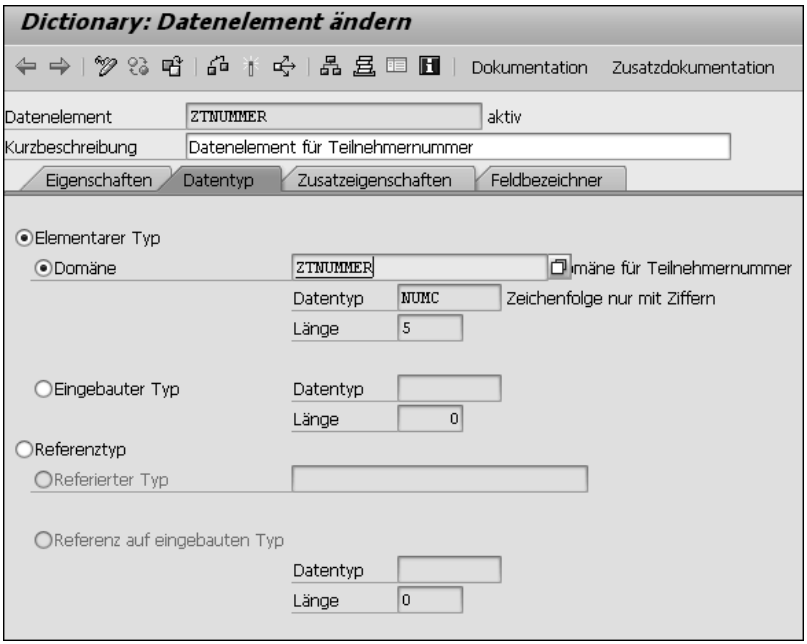


Abbildung 2.8 Datenelement pflegen

Auf der Registerkarte **Datentyp** geben Sie die verknüpfte Domäne an. Denken Sie daran, dass Datenelemente und Domänen verschiedene Objekttypen sind. Sie können deshalb für die neu anzulegende Domäne wieder einen anderen Namen erfinden, aber auch denselben Namen wie für das Datenelement verwenden.

Im Übrigen ist wiederum nur der Kundennamensraum zu beachten. Durch den Objekttyp **Domäne** hat das System kein Problem bei der Identifizierung des richtigen Objekts. In unserem Beispiel soll die Domäne zur Vereinfachung und Vereinheitlichung auch ZTNUMMER heißen. Tragen Sie die Domäne in das entsprechende Feld ein, und sichern Sie das Datenelement.

Zunächst müssen Sie wieder eine **Kurzbeschreibung** des Objekts eingeben, z. B. »Datenelement für Teilnehmernummer«. Da das Datenelement ein eigenes Objekt ist, möchte das System wissen, ob das Objekt transportiert werden soll oder nicht. Sie müssen deshalb wieder einen Paketnamen angeben. Wie zuvor transportieren Sie Ihre ersten Gehversuche aber nicht und speichern das Datenelement daher als **Lokales Objekt** ab (siehe Abbildung 2.9).

Abbildung 2.9 Datenelement sichern

Auf der Registerkarte **Feldbezeichner** müssen Sie als Nächstes die Bezeichnungen des Felds angeben, die später in Dialogprogrammen und Listen ausgegeben werden. In Bildschirmmasken späterer Dialogprogramme können Sie das Tabellenfeld z. B. mit seinem Kurz-, Mittel- oder Langtext einbauen; in Listen erhält die Spalte für das Feld eine Überschrift. Alle Texte hierzu pflegen Sie an dieser Stelle und füllen dazu die entsprechenden Felder der Spalte **Feldbezeichner** mit den Angaben aus Abbildung 2.10. Nachdem Sie Ihre Eingaben gesichert haben, rechnet das System die Textlängen aus und füllt die Werte der Spalte **Länge**. Die mindestens notwendigen Voraussetzungen, um ein Datenelement anzulegen, sind somit erfüllt.

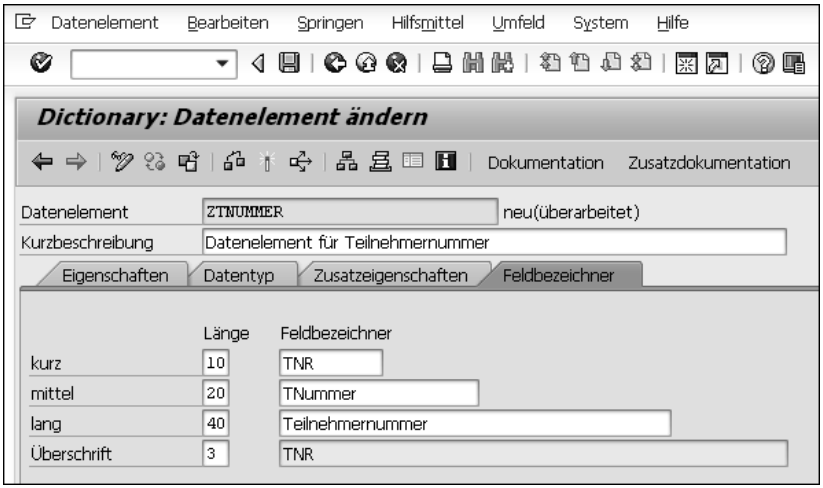


Abbildung 2.10 Feldbezeichner pflegen

Nach einem Klick auf den Button **Dokumentation** (in der Funktionsleiste in Abbildung 2.10 zu sehen) wird Ihnen der Bildschirm eines anderen SAP-Werkzeugs angezeigt: des *SAPscript-Editors*. SAPscript ist ein schon in die Jahre gekommenes, aber unverwüstliches Tool. Neben seinem Einsatz in der Onlinedokumentation wird es in der SAP-Welt unter anderem für die Formulargestaltung verwendet und verfügt über einen eigenen Editor und einen eigenen Kommandosatz. Klicken Sie einfach in den Abschnitt **&DEFINITION&**, und schreiben Sie den Hilfetext »Bitte geben Sie eine Teilnehmernummer zwischen 1 und 99999 ein.« in die Zeile unterhalb des Eintrags **&DEFINITION&** (siehe Abbildung 2.11). Sichern Sie Ihre Eingabe, und kehren Sie über die F3-Taste zur Pflege des Datenelements zurück.

Dokumentation

Da die Erfahrung zeigt, dass man sich als Entwickler ebenso wie als Anwender zu oft über schlecht dokumentierte Programme anderer Entwickler ärgert, werden wir es an dieser Stelle natürlich besser machen und legen einen erklärenden Text für unser Datenelement an. Falls später jemand nicht weiß, wofür dieses Feld gedacht ist – obwohl es z. B. vom Namen her für ein Projekt sinnvoll erscheint –, erhält die Person eine sachliche Erklärung und Hilfestellung für die Eingabe oder Verwendung in eigenen Programmen.

Wie alle Texte wird auch die Dokumentation sprachabhängig in der Anmeldesprache gepflegt, in unserem Beispiel in Deutsch. Bei Bedarf werden die Texte und Dokumentationen mittels eines separaten Werkzeugs in die Zielsprache übersetzt.

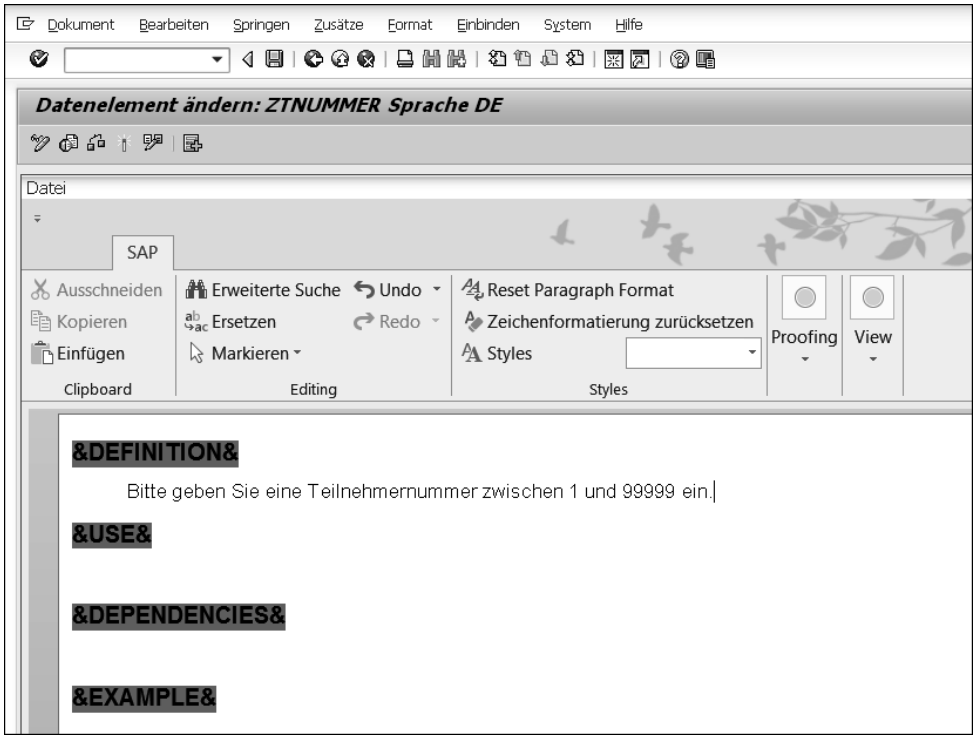


Abbildung 2.11 Hilfetext pflegen

2.2.2 Domäne anlegen

Auf dem Pflegebildschirm des Datenelements haben Sie auf der Registerkarte **Datentyp** bereits den Domänennamen angegeben. Nachdem die Arbeit mit dem Datenelement vorerst abgeschlossen ist, können wir diese Domäne anlegen. Am besten arbeiten Sie dabei wieder mit der Vorwärtsnavigation, das heißt einem Doppelklick auf den Domänennamen `ZTNUMMER`.

Sie werden nun gefragt, ob Sie die Domäne anlegen möchten, und beantworten die Frage mit **Ja**. Daraufhin wird Ihnen der Pflegebildschirm für die Domäne angezeigt wie Sie in Abbildung 2.12 sehen. Bevor wir uns den Registerkarten zuwenden, müssen Sie wieder eine Kurzbeschreibung eingeben, beispielsweise »Domäne für Teilnehmernummer«.

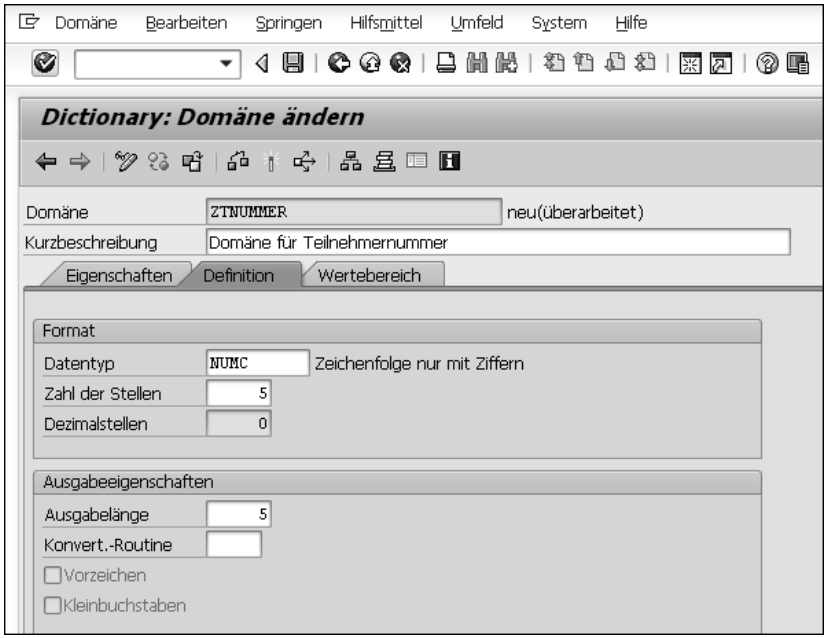


Abbildung 2.12 Domäne pflegen

Auf der Registerkarte **Definition** müssen Sie zunächst den Datentyp der Domäne angeben. In der Praxis sollten Sie viel früher, bereits bei der Konzeption der Tabelle »auf dem Papier«, alle Attribute des Felds festlegen. An dieser Stelle können Sie aber über die F4-Taste sehen, welche Datentypen im ABAP Dictionary enthalten sind und über welche Eigenschaften sie jeweils verfügen. Für unser Beispiel wählen Sie als Datentyp `NUMC`, das heißt eine Zeichenfolge nur mit Ziffern und der Ausgabelänge 5. Anschließend sichern Sie das Domänenobjekt wieder als **Lokales Objekt**.

Nach dieser Deklaration der Domäne könnte es zusätzlich sinnvoll sein, die gültigen Werte für das Tabellenfeld einzugrenzen. Auf dem Pflegebildschirm für die Domäne existiert hierzu die Registerkarte **Wertebereich** (siehe Abbildung 2.13).

Grundsätzlich sind hier drei verschiedene Eingrenzungen möglich: Einzelwerte, Intervalle oder eine Wertetabelle. Falls Sie in diese Felder etwas eintragen, akzeptiert das System nur noch die hier definierten und damit gültigen Werte; alle anderen Eingaben werden abgewiesen. Beachten Sie jedoch, dass die Reaktion des Systems vom Datentyp abhängig ist.

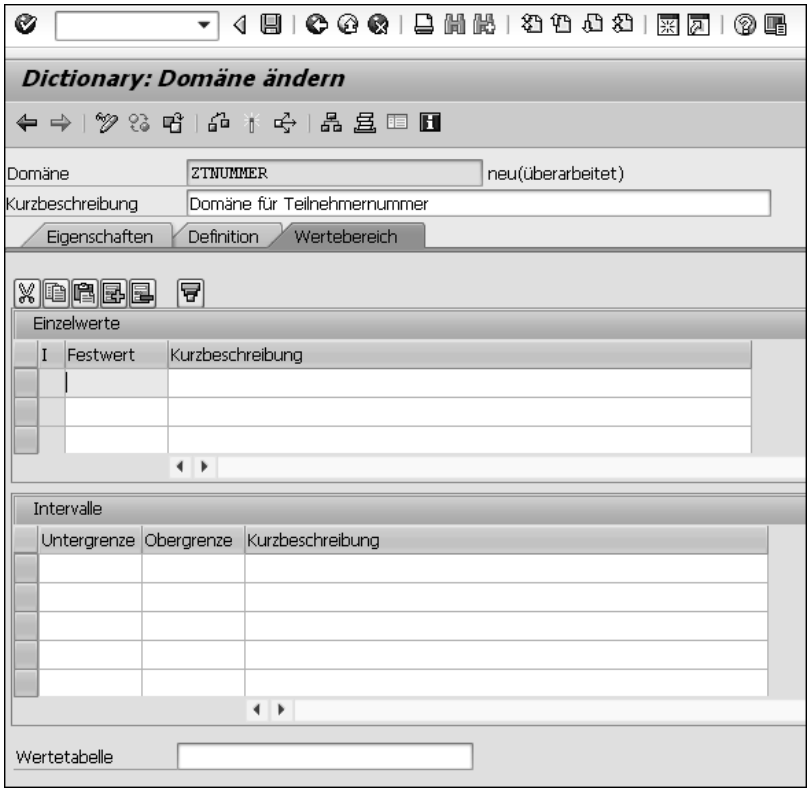


Abbildung 2.13 Wertebereich einer Domäne festlegen

So können *Festwerte*, das heißt Einzelwerte und Intervalle, nicht für alle Datentypen eingetragen werden, und auch die Eingabeprüfung bei Dialogbildschirmen hängt vom Datentyp ab:

- **Einzelwerte**
  Bei der Pflege von Einzelwerten wird jeder gültige Wert einzeln aufgeführt und beschrieben. Die Pflege von Einzelwerten in der Domäne ist nur bei einer relativ kleinen Anzahl zulässiger Eingaben sinnvoll.
- **Intervalle**
  Pflegt man in der Domäne gültige Intervalle ein, wird jeweils der kleinste und der größte Wert eines Intervalls angegeben. Alle Werte dazwischen sind ebenfalls gültig, einschließlich der Grenzwerte.
- **Wertetabelle**
  Das Arbeiten mit einer Wertetabelle ist bei großen Datenmengen zu empfehlen, wenn absehbar ist, dass Eingabeprüfungen für das Feld anhand einer entsprechenden

Tabelle erfolgen sollen. Beachten Sie, dass nur durch das Eintragen der Wertetabelle noch keine Prüfung eingerichtet wird. Hierzu muss erst noch ein *Fremdschlüssel* für das Tabellenfeld implementiert werden. Auf die Arbeit mit Fremdschlüsseln gehen wir in Abschnitt 7.2.3, »Fremdschlüssel pflegen«, ein.

In diesem Beispiel verzichten wir darauf, Festwerte oder Wertetabellen einzutragen, und sichern die Domäne. Um formale Fehler auszuschließen, sollten Sie im nächsten Schritt die Domäne prüfen, entweder über den Menüpfad **Domäne • Prüfen • Prüfen** (siehe Abbildung 2.14), über die Tastenkombination Strg+F2 oder über den **Prüfen**-Button ( ). Im günstigen Fall erhalten Sie die Meldung, dass das System keine Inkonsistenzen gefunden hat, und können dazu übergehen, die Domäne zu aktivieren.

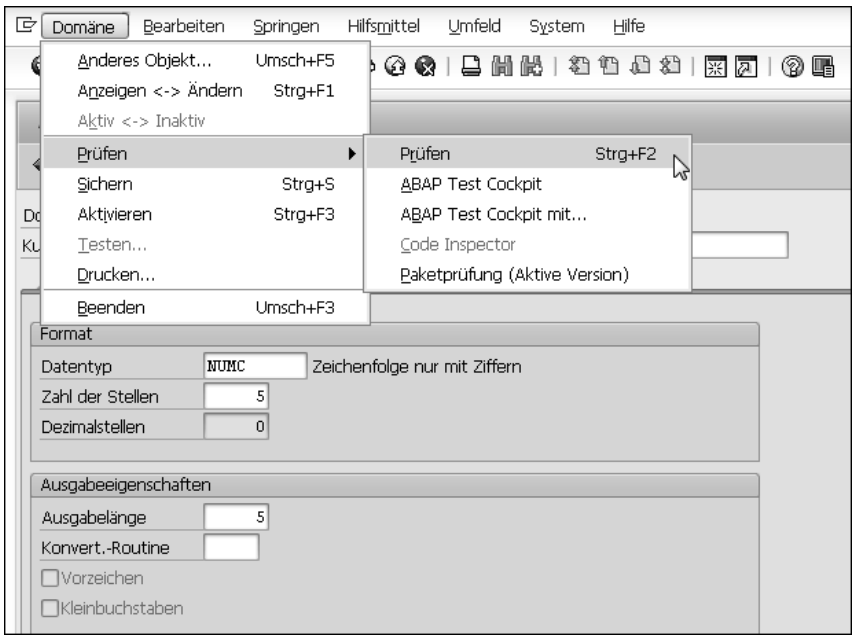


Abbildung 2.14 Domäne prüfen

Noch hat die Domäne den Status **neu (überarbeitet)** (siehe Abbildung 2.15). Um sie in einem Datenelement verwenden zu können, muss sie den Status **aktiv** haben – dies erreichen Sie, indem Sie die Domäne aktivieren. Auch hierfür gibt es wieder die Möglichkeiten, über das Menü zu navigieren (**Domäne • Aktivieren**), eine Tastenkombination zu verwenden (Strg+F3) oder den **Aktivieren**-Button ( ) anzuklicken.

Abbildung 2.15 Domäne aktivieren

Im folgenden Dialogfenster erhalten Sie eine Übersicht über alle momentan inaktiven Objekte. Die Domäne ist bereits markiert (siehe Abbildung 2.16). Bei Bedarf können Sie die anderen, inaktiven Objekte ebenfalls markieren und aktivieren. Falls die Arbeit an diesen Objekten aber noch nicht abgeschlossen ist, sollten Sie keine unnötigen Risiken eingehen und wie geplant nur die gewünschte Domäne aktivieren.

Abbildung 2.16 Objekte aktivieren

Bestätigen Sie Ihre Auswahl über den Button mit dem grünen Häkchen rechts unten bzw. über die [↵]-Taste. Sie erhalten unten in der Statuszeile die Meldung, dass das Objekt aktiviert wurde. Zur Kontrolle können Sie auch den Status vergleichen, der jetzt auf **aktiv** steht (siehe Abbildung 2.17). Die Domäne ist damit systemglobal verwendbar. Mit der Taste [F3] verlassen Sie den Pflegebildschirm und kehren zum Datenelement zurück.

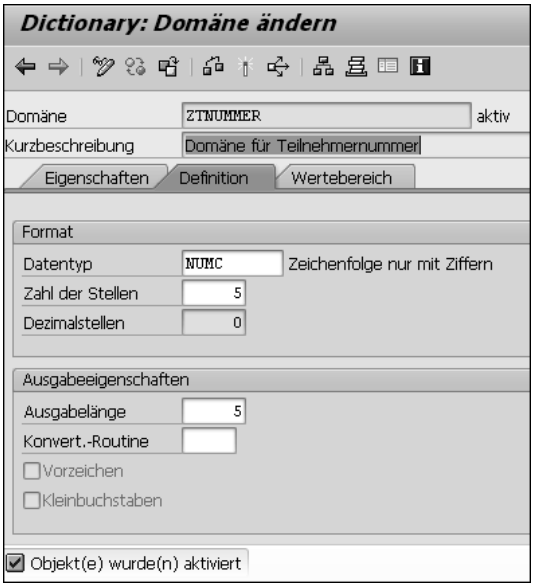


Abbildung 2.17 Aktivierte Domäne

2.2.3 Datenelement prüfen und aktivieren

Zum Einstieg sollten Sie auch vor dem Aktivieren des Datenelements eine Konsistenzprüfung durchführen. Hierzu arbeiten Sie wieder mit dem Menü, der Tastenkombination oder dem **Prüfen**-Button (, siehe Abbildung 2.18). Im Idealfall haben Sie alles richtig gemacht und können mit der Aktivierung des Datenelements fortfahren.

Abbildung 2.18 Datenelement prüfen

Dieser Arbeitsschritt entspricht dem beschriebenen Vorgehen für die Domäne: Aktivieren Sie das Datenelement über das Menü oder den **Aktivieren**-Button (), bestätigen Sie im Folgedialog die Aktivierung des Datenelements, und prüfen Sie, ob die Aktivierung erfolgreich war, das heißt, ob der Status **aktiv** für das Datenelement und die entsprechende Meldung in der Statuszeile angezeigt werden. Anschließend kehren Sie mit der Taste F3 zur Bearbeitung der Tabellenfelder zurück.

Um in der Tabelle sinnvolle Datensätze ablegen zu können, sollte diese für unser Beispielszenario um weitere Felder ergänzt werden. Es bleibt Ihnen selbst überlassen, wie viele Felder Sie – zur Übung – in die Tabelle einpflegen möchten. Für das weitere Vorgehen genügen vorerst vier Felder, um einen kleinen Datensatz in die Tabelle einzupflegen. Das Ergebnis sollte dann wie in Abbildung 2.19 aussehen.

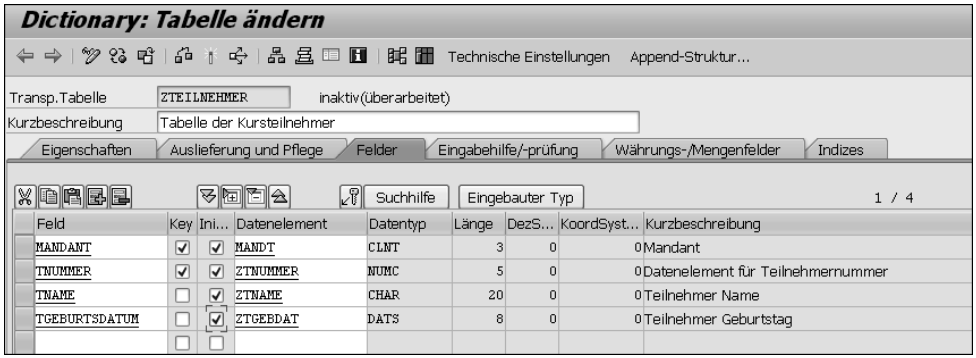


Abbildung 2.19 Beispieltabelle mit vier Feldern

Tabelle prüfen

Vergessen Sie nicht, auch die Tabelle über den **Prüfen**-Button (), über die Tastenkombination Strg+F2 oder das Menü auf Inkonsistenzen hin zu überprüfen. Diesen Schritt sollten Sie sich von Anfang an angewöhnen, und er sollte Ihnen in Fleisch und Blut übergehen.

2.2.4 Technische Einstellungen der Tabelle pflegen

Noch liegt die Tabelle lediglich im ABAP Dictionary. Deshalb müssen Sie vor der Aktivierung noch die technischen Einstellungen pflegen. Sie legen dort Speicherparameter fest und bestimmen, ob und wie Datensätze beim Lesen aus der Datenbank in einem Lesepuffer gehalten werden sollen.

Zu den technischen Einstellungen navigieren Sie aus dem Tabellenpflegebildschirm des ABAP Dictionary über den Menüpfad **Springen • Technische Einstellungen** oder über den Button **Technische Einstellungen**. Es erscheint der entsprechende Pflegebildschirm (siehe Abbildung 2.20).

In unserem Beispiel verwenden wir für die **Datenart** den Wert APPLO (Stammdaten in transparenten Tabellen). Sie legen damit bereits den richtigen physischen Bereich für die Tabelle auf der Datenbank fest. Der physische Bereich, auch *Tablespace* genannt, ist

je nach Konfiguration des Datenbanksystems eine Zuordnung zu einem physischen Ablageort wie z. B. einem Verzeichnis oder Laufwerk der Datenbank.

Abbildung 2.20 Technische Einstellungen der Tabelle pflegen

Je nach Datenbanksystem, Datenvolumen und der Anzahl von Zugriffen beeinflusst diese Entscheidung das spätere Laufzeitverhalten der Tabelle und soll die Lese- und Schreibzugriffe optimieren. Stammdaten oder Customizing-Daten für die Systemeinstellungen können in jeweils eigenen physischen Datenbankbereichen abgespeichert werden. Auch das Pufferungsverhalten, das das System beim Zugriff auf die Tabellen an den Tag legt, kann sich je nach Datenart unterscheiden. Insgesamt handelt es sich um Optimierungen, die das System vornimmt und bei denen Sie es unterstützen, indem Sie Ihre Daten mithilfe der Datenart kategorisieren.

Im Feld **Größenkategorie** geben Sie »0« ein, da wir für dieses Übungsbeispiel nicht viele Datensätze erwarten. Die Größenkategorie legt die Größe des initialen Speicherplatzes für die Tabelle auf der Datenbank fest. Wählen Sie den Platz zu groß, reservieren Sie Bereiche, die ungenutzt bleiben; wählen Sie den Platz zu klein, müssen Sie häufiger reorganisieren.

Kleinste Kategorie

Im Hilfefenster zu den Größenkategorien erscheinen Beispielzahlen für die zu erwartenden Datensätze. Lassen Sie sich durch diese Zahlen nicht irritieren: Diese Vorschlagswerte sollen Ihnen einfach einen Anhaltspunkt geben, welche Größenordnungen als klein, mittel, groß etc. gelten dürfen. Für unser Beispiel reicht einfach die kleinste der vier Größenkategorien aus.

Für die Pufferungsart wählen Sie **Pufferung nicht erlaubt**. Bei der Pufferung geht es allgemein darum, dass die Tabelle zum Lesen ganz oder teilweise in den Arbeitsspeicher geladen wird – es wird sozusagen »auf Vorrat« gelesen. Aus Performancegründen empfiehlt sich die Pufferung von Tabellen oder Datensätzen bei großen Tabellen mit vielen Lesezugriffen und relativ wenigen Schreibzugriffen. Bei bestimmten Leseoperationen, beispielsweise wenn Summen gebildet werden, wird die Pufferung in jedem Fall umgangen, und es erfolgt ein Lesezugriff auf die Datenbank, anstatt die Anfrage aus dem Puffer des Applikationsservers zu beantworten. Bei unserem Übungsbeispiel arbeiten wir jedoch nur mit einer sehr kleinen Tabelle und wenigen Zugriffen, sodass das Pufferungsverhalten nicht ins Gewicht fällt.

2.2.5 Erweiterungskategorie pflegen

Beim Anlegen einer Tabelle oder einer anderen Dictionary-Struktur sollten Sie stets die *Erweiterungskategorie* des Objekts pflegen. Diese dient vor allem bei Objekten des SAP-Standards dazu festzulegen, welche Erweiterungen nachträglich an diesem Dictionary-Objekt vorgenommen werden können, ohne dass die Programme, die damit arbeiten, in ihrer Funktionsweise beeinträchtigt werden.

Beispielsweise könnte eine Datenstruktur des SAP-Standards in Programmen verwendet werden, die zwei weiterhin funktionieren, wenn ein Kunde mittels der angebotenen Erweiterungstechniken (mehr dazu in Kapitel 7, »Transparente Datenbanktabellen bearbeiten«) weitere Felder anhängt; dies aber nur dann, wenn diese Felder zeichenartig sind. Eine solche Datenstruktur würde mit der Erweiterungskategorie **erweiterbar und zeichenartig** gekennzeichnet.

In unserem Fall wollen wir keine solchen Einschränkungen festlegen und wählen daher die Erweiterungskategorie **beliebig erweiterbar**, wie in Abbildung 2.21 dargestellt.

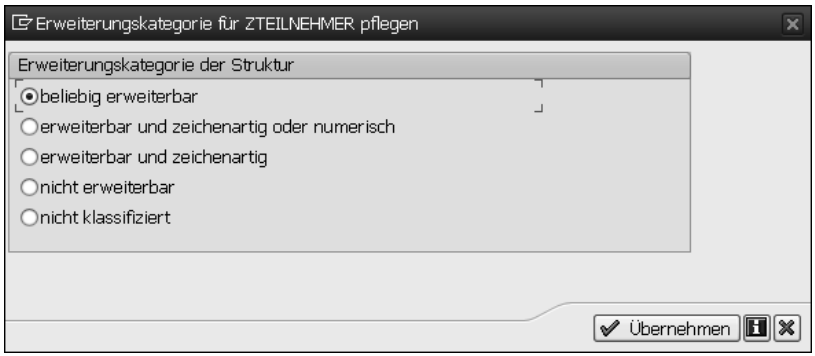


Abbildung 2.21 Erweiterungskategorie auswählen

Nach diesen Einstellungen kehren Sie mit der Taste F3 zur Tabellenpflege zurück und aktivieren die Tabelle. Den Erfolg erkennen Sie an die Meldung in der Statuszeile (»Objekt wurde aktiviert«), aber auch am Status der Tabelle, der jetzt auf **aktiv** steht (siehe Abbildung 2.22).

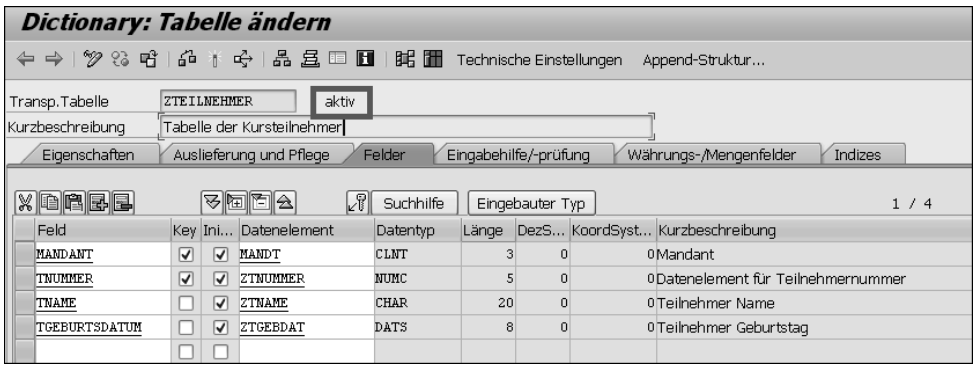


Abbildung 2.22 Aktivierte Tabelle

2.3 Datensätze erfassen

Bereits jetzt können Sie aus dem Pflegedialog heraus Datensätze für die Tabelle erfassen. Um einige Datensätze in die Tabelle zu schreiben, navigieren Sie über den Menüpfad **Hilfsmittel • Tabelleninhalt • Einträge erfassen** (siehe Abbildung 2.23) in einen Pflegedialog, in dem Sie Datensätze erfassen und sichern können (siehe Abbildung 2.24).

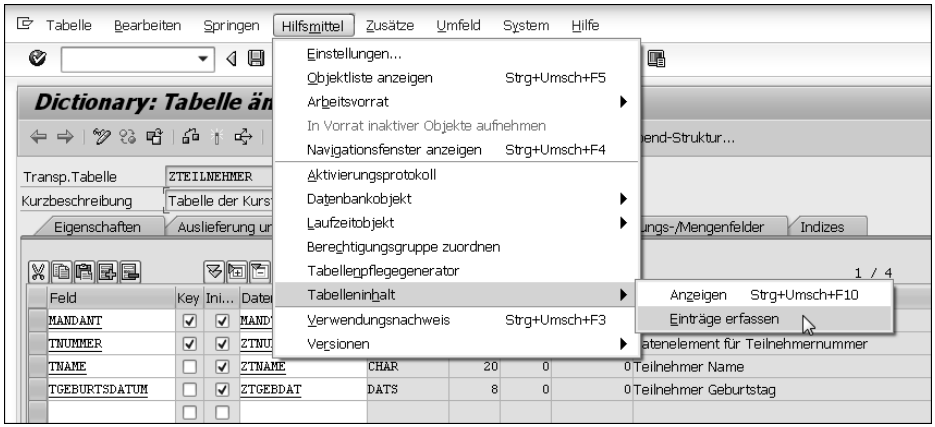


Abbildung 2.23 Einträge in Tabelle erfassen

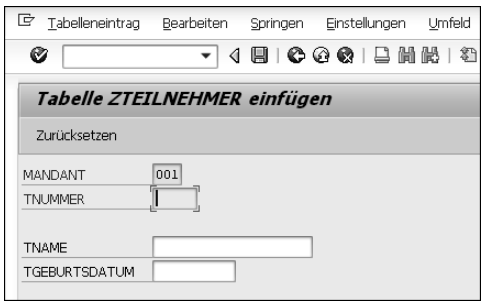


Abbildung 2.24 Pflegedialog zum Erfassen von Datensätzen

2.3.1 Datensätze eingeben

Zum Testen geben Sie drei Datensätze ein, jeweils eine Teilnehmernummer (im Feld **TNUMMER**), einen Teilnehmernamen (im Feld **TNAME**) und den Geburtstag des Teilnehmers (im Feld **TGEBURTSDATUM**). Das Feld **MANDANT** ist in der Hintergrundfarbe hinterlegt und nicht eingabebereit, da die automatische Mandantenbehandlung, auf die wir in Abschnitt 9.3, »Open-SQL-Anweisungen«, noch einmal zurückkommen werden, in dieses Feld selbstständig den Anmeldemandanten einträgt. Schlüsselfelder, in unserem Beispiel die Teilnehmernummer, sind gelb hinterlegt, die anderen Felder weiß. Sichern Sie über den **Sichern**-Button (💾) jeden einzelnen Datensatz. In der Statuszeile erhalten Sie nach dem Sichern jedes Satzes die Meldung »Datensatz erfolgreich weggeschrieben«.

Zur Kontrolle, ob die Dokumentation beim Datenelement sauber eingebaut wurde, können Sie bereits hier das Eingabefeld für das Geburtsdatum fokussieren, das heißt anklicken, und die Taste F1 drücken. Haben Sie alles richtig gemacht, erscheint in einem

neuen Fenster die betriebswirtschaftliche Hilfe (*F1-Hilfe*). Falls Sie Festwerte in der Domäne verankert haben, lässt sich das über die Wertehilfe prüfen, die Sie mit der Taste F4 aufrufen. Anschließend kehren Sie über den **Zurück**-Button (🔙) oder die Taste F3 in den Pflegedialog der Tabelle zurück.

2.3.2 Tabelleninhalt anzeigen

Aus dem Pflegedialog heraus können Sie sich die Datensätze der Tabelle zur Prüfung ansehen. Navigieren Sie hierzu, wie eben gezeigt, über den Pfad **Hilfsmittel • Tabelleninhalt**, wählen Sie nun aber **Anzeigen**. Um bestimmte Sätze zu selektieren, könnten Sie nun die Auswahl eingrenzen. Sie möchten sich aber alle Datensätze ansehen – das heißt die, die Sie gerade eingepflegt haben – und klicken daher direkt auf den **Ausführen**-Button (🕒, siehe Abbildung 2.25).

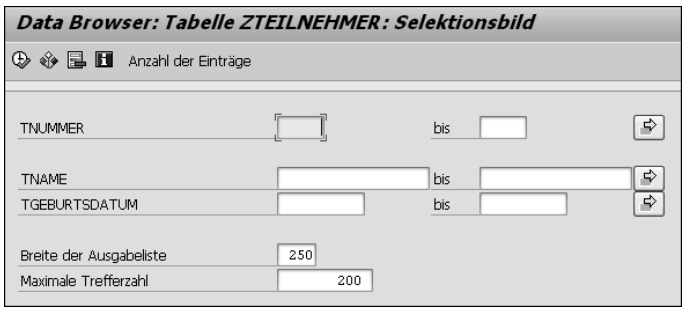


Abbildung 2.25 Tabelleninhalt anzeigen

Wenn Sie nun eine Liste wie in Abbildung 2.26 sehen, hat alles wie gewünscht geklappt. Natürlich sind Tabellen in der Praxis ein wenig umfangreicher, und viele Möglichkeiten des ABAP Dictionary haben wir bislang noch nicht erläutert. Aber für den Einstieg in ABAP liegt das Wesentliche bereits hinter Ihnen: Sie haben eine Tabelle angelegt, mit Daten gefüllt und können sich ab sofort mit voller Kraft der ABAP-Programmierung zuwenden.

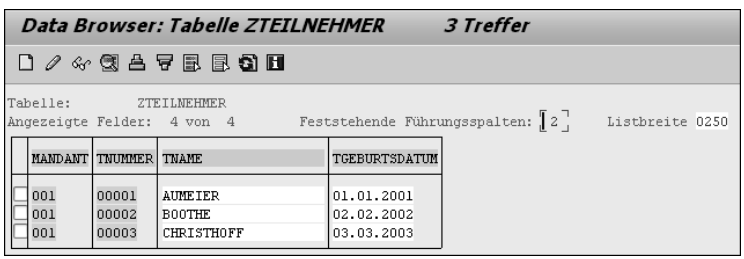


Abbildung 2.26 Datensätze anzeigen

Einleitung

Dieses Buch ist auch in der 6. Auflage ein Lehr- und Lernbuch für alle Interessierten, die in die Welt der ABAP-Programmierung einsteigen möchten. Diesem Ziel ordnet es einiges unter: An manchen Stellen »lügt« das Buch, indem es vorgibt, alles sei ganz unkompliziert. Es vereinfacht, indem es technische und betriebswirtschaftliche Probleme auf ein Minimum reduziert. Es nimmt gewisse Lücken in Kauf, weil es das Thema ABAP nicht in epischer Breite behandeln will, sondern sich auf einen thematischen Kern und den roten Faden zum Lernziel beschränkt. Denn dieses Buch möchte Ihnen vor allem eines vermitteln: Erfolgserlebnisse.

»Einstieg in ABAP« kann keinen allgemeinen SAP-Grundlagenkurs ersetzen, ebenso wenig wie vertiefende Literatur zu speziellen ABAP-Themen wie den neuen, gerade im Entstehen begriffenen cloudbasierten ABAP-Varianten. Das Buch erhebt auch keinen Anspruch auf Vollständigkeit. Es möchte kein Trockenkurs sein, sondern Sie zum Lernen am und zum Arbeiten im SAP-System motivieren. Alle hierfür relevanten Arbeitsschritte werden ausführlich mit Beispielen, Hintergrundinformationen und Quellcodebeschreibungen erklärt.

An wen richtet sich dieses Buch?

Dieses Buch ist für Entwickler geschrieben, in deren Firmen ein SAP-System (on premise, d. h. vom Kunden selbst verwaltet und betrieben) eingeführt wird. Es richtet sich an Berater und Projektleiter, die ABAP-Quellcode lesen und verstehen sowie einige Änderungen selbst vornehmen möchten. Außerdem ist es für Studenten und Auszubildende geschrieben, die ABAP im Rahmen ihrer Berufsausbildung erlernen. Besondere Vorkenntnisse sind nicht erforderlich, auch wenn sich einzelne Abschnitte an Programmierer mit Vorerfahrungen aus anderen Programmiersprachen wenden.

Über dieses Buch

Sie werden vom einfachsten ABAP-Report bis hin zur modularisierten Ablaufsteuerung geführt. Auch wichtige Arbeiten zur Tabellenpflege im ABAP Dictionary und komplexe Datenübergabestrukturen zwischen Reports kommen nicht zu kurz. Außerdem lernen Sie die datenbanknahe Programmierung mit CDS Views kennen. Hierbei wird ein einfaches betriebswirtschaftliches Beispiel, eine fiktive Anwendung zur Verwaltung der Teilnehmer von IT-Kursen, durchgängig mitgeführt und immer wieder verwendet, um die Theorie unmittelbar in der Praxis anzuwenden. Der Schwerpunkt dieses Buches liegt

demnach auf der Vermittlung und Aneignung von ABAP-Wissen und nicht auf betriebswirtschaftlichen Zusammenhängen und Prozessen.

Neuerungen in der sechsten Auflage

Die SAP-Welt dreht sich schneller denn je weiter, und davon sind die Programmiersprache ABAP und ihre natürliche Umgebung, der ABAP-Applikationsserver, in besonders starkem Maße betroffen. Auch wenn viele Weiterentwicklungen eher fortgeschrittene Programmierer betreffen und in diesem Buch nicht vertieft betrachtet werden können, haben wir ihnen so gut wie möglich Rechnung getragen und Verweise auf aktuelle Themen wie SAP Fiori und SAP HANA an passenden Stellen eingefügt. Vielleicht möchten Sie auf dieser Basis auf eigene Faust bzw. zu einem späteren Zeitpunkt über den Tellerrand des ABAP-Einsteigers schauen.

Die Gemeinde der ABAP-Programmierer erlebt heute erfreulicherweise viel Zuwachs von Programmierern, die Erfahrungen mit den unterschiedlichsten Programmiersprachen mitbringen. Waren es früher vor allem erfahrene COBOL-Programmierer und Geschäftsprozessexperten ohne vorherige Programmiererfahrung, die die Gemeinde bereicherten, kommen heute Webentwickler mit Kenntnissen der neuesten Frontend- und Backend-Technologien, Java-Entwickler, Entwickler mit Erfahrung in den modernen Skriptsprachen sowie in cloudbasierten Programmiermodellen und viele andere hinzu. Die ABAP-Welt ist heute viel weniger in sich geschlossen als noch vor einigen Jahren. Dieser Vielfalt der benachbarten Technologien ist es geschuldet, dass in diesem Buch eine Vielzahl von Bezügen zu Themen auftauchen, auf die wir nicht vertieft eingehen können. In diesem Fall vermitteln wir Ihnen die Grundlagen zur ersten Einordnung des Themas und verweisen auf andere Bücher, die das jeweilige Thema ausführlich behandeln.

Wie können Sie mit diesem Buch arbeiten?

In diesem Buch finden Sie mehrere Orientierungshilfen. Die folgenden Symbole helfen Ihnen dabei, sich schneller zu orientieren:

Tipp: Dieses Symbol steht an Stellen, die spezielle Tipps und Empfehlungen bereithalten, die Ihnen die Arbeit erleichtern können.

Hinweis: Hinweise geben Informationen zu weiterführenden Themen, zu weiteren Quellen oder wichtigen Inhalten, die Sie sich merken sollten.

Achtung: Dieses Symbol warnt Sie vor Fallstricken und typischen Fehlern.

Beispiel: Unter diesem Symbol finden Sie Szenarien und Beispiele aus der Praxis.

Systemvoraussetzungen

Um von diesem Buch optimal profitieren zu können, sollten Sie einige Voraussetzungen mitbringen: Ideal wäre es sicherlich, sich mit SAP-Grundlagen wie der Navigation im Hauptmenü schon vorab beschäftigt zu haben. Auch Erfahrungen mit allgemeiner Programmierlogik aus anderen Programmiersprachen, mit Makros oder Skripten sind für den Lernerfolg förderlich. Ideal wäre auch der Zugang zu einem SAP-System mit den entsprechenden Berechtigungen – hier genügen zum großen Teil aber auch die SAP-Trial-Systeme.

SAP-Trial-Systeme

Bei den SAP-Trial-Systemen, auch ABAP-Trial-Systeme genannt, handelt es sich um kostenlose, von SAP zu Lehr- und Testzwecken bereitgestellte, persönlich nutzbare ABAP-Systeme. Den einfachsten Einstieg bietet hier die *SAP Cloud Appliance Library* (*http://cal.sap.com*). Auf dieser Webseite bietet SAP eine Vielzahl von Trial-Versionen für viele unterschiedliche SAP-Softwareprodukte. Einige davon sind kostenpflichtig, doch die meisten können nahezu kostenlos bezogen werden.

Diese Testsysteme laden Sie nicht wie das sogenannte *Mini-SAP-System* in früheren Releases herunter und installieren es auf Ihrem eigenen Rechner, sondern nutzen Cloud-Dienstleister wie Amazon Web Services oder Microsoft Azure, um Ihr persönliches Testsystem in der Cloud Ihrer Wahl zu installieren.

Ein Hinweis, um böse Überraschungen zu vermeiden: Durch die Nutzung des Cloud-Dienstleisters entstehen Ihnen Kosten, die Sie sorgfältig kontrollieren sollten, wenn Sie ein solches System nutzen. Studieren Sie die Preisliste, damit Sie wissen, welcher Betrag für die Nutzung auf Sie zukommt. Normalerweise erfolgt die Bezahlung pro Stunde, in der das System aktiv ist. Achten Sie daher stets darauf, Ihr System herunterzufahren, wenn Sie für den Tag damit fertig sind, und löschen Sie es, wenn Sie es nicht mehr benötigen.

Wenn Sie ein lokal installiertes System bevorzugen, prüfen Sie im *SAP Developer Center* (*https://developers.sap.com/*) oder im *SAP Store* (*https://www.sapstore.com/*), ob aktuell ein Trial-System zur lokalen Installation verfügbar ist. Zum Zeitpunkt der Drucklegung dieses Buchs bietet SAP keine Möglichkeit zum Download für die lokale Installation, aber da es gerade seitens der Open-Source-Community einen großen Bedarf an lokal installierbaren ABAP-Systemen gibt, könnte sich das in naher Zukunft auch wieder ändern.

Ob lokal oder in der Cloud: Das ABAP-Trial-System stellt die ABAP-Entwicklungsumgebung bereit – mehr benötigen Sie für den Lernerfolg mit diesem Buch nicht.

Die Beispiele und Abbildungen in diesem Buch beziehen sich auf SAP ABAP Platform 1909 (Basis-Release 7.54), das zum Zeitpunkt der Drucklegung neueste Release eines rei-

nen ABAP-Systems ohne betriebswirtschaftliche Anwendungen als separat zu installierende Software. Sie können aber auch problemlos mit älteren oder neueren Releases genutzt werden, wie sie Ihnen beispielsweise in Projekten mit SAP S/4HANA oder SAP Business Warehouse begegnen werden.

Das Release SAP ABAP Platform 1909 markiert eine wichtige Änderung für die Standalone-ABAP-Installation, die früher den Namen SAP NetWeaver AS ABAP trug: Es wurde bei der Weiterentwicklung zur *ABAP Platform* zugleich die separate Installierbarkeit ohne die Anwendungssoftware SAP S/4HANA aufgegeben. Das neueste Release der ABAP Platform, das also nur noch als Bestandteil einer SAP-S/4HANA-Installation vorkommt, trägt die Releasebezeichnung *ABAP Platform 2022* und ist Bestandteil des Produkts SAP S/4HANA 2022.

Kapitelübersicht und Aufbau

Kapitel 1, »ABAP und die ersten Schritte im SAP-System«, vermittelt Ihnen das für den Einstieg in ABAP notwendige Wissen über die organisatorische und technische Architektur von SAP-Systemen und die Entwicklungsvoraussetzungen. Ferner wird die Arbeitsteilung zwischen der Laufzeitumgebung und den Anwendungsprogrammen gezeigt sowie die Struktur von ABAP-Reports.

Kapitel 2, »ABAP Dictionary«, liefert einen Überblick über den Sinn und Zweck des ABAP Dictionarys. Am Beispiel einer transparenten Tabelle erlernen Sie das Anlegen einer Tabelle bis hin zum Erfassen und Anzeigen von Tabelleneinträgen und alle notwendigen Arbeiten mit Datenelementen, Domänen und technischen Einstellungen.

In **Kapitel 3**, »Programmieren im ABAP Editor«, legen Sie Ihren ersten ABAP-Report an, pflegen dessen Eigenschaften, erstellen den Quellcode und führen diesen aus. Gleichzeitig lernen Sie die ersten ABAP-Anweisungen und die notwendige Syntax kennen.

Das Grundrechnen mit Variablen üben Sie in **Kapitel 4**, »Felder und Berechnungen«; Sie betrachten die Eigenschaften von Datenobjekten und den Unterschied zwischen Kompatibilität und Konvertibilität. Gleichzeitig lernen Sie einige Kniffe für eine bessere Listaufbereitung kennen.

In **Kapitel 5**, »Mit Zeichenketten arbeiten«, arbeiten Sie mit Zeichenketten und führen String-Operationen in verschiedenen Varianten aus. Sie suchen Teil-Strings und modifizieren Zeichenketten.

Kapitel 6, »Debugging von Programmen«, ist dem Verfolgen des Programmablaufs und dem Suchen nach Programmfehlern gewidmet. Sie arbeiten zu diesem Zweck mit dem ABAP Debugger und seinen wichtigsten Modi. Das schichtenorientierte Debugging wird in diesem Rahmen vorgestellt.

In **Kapitel 7**, »Transparente Datenbanktabellen bearbeiten«, erweitern und modifizieren Sie eine transparente Tabelle, lernen, was ein Fremdschlüssel ist, und bauen Prüfungen für die Tabelleninhalte ein. Hierzu arbeiten Sie mit Festwerten in der Domäne, mit Wertetabellen und Prüftabellen. Außerdem lernen Sie Besonderheiten bei Währungs- und Mengenfeldern kennen.

In **Kapitel 8**, »Rechnen mit Datum und Zeit, Mengen und Währungen«, deklarieren Sie Datums- und Zeitfelder, verarbeiten diese und betrachten deren besondere Eigenschaften. Dem Verständnis dienen viele Beispiele, die auch im ABAP Debugger verfolgt und beschrieben werden.

Den Inhalt einer Datenbanktabelle modifizieren Sie in **Kapitel 9**, »Mit Daten in einer Datenbanktabelle arbeiten«. Sie fügen neue Zeilen über ABAP-Anweisungen ein und ändern und löschen bestehende Zeilen. Auch die Risiken beim Löschen und der Datenmanipulation werden gezeigt.

Das Anwendungsbeispiel wird komplexer: In **Kapitel 10**, »Programmablaufsteuerung und logische Ausdrücke«, treffen Sie Fallunterscheidungen, bauen Kontrollstrukturen und Verzweigungen ein und lernen logische Ausdrücke kennen.

In **Kapitel 11**, »Selektionsbildschirme«, lernen Sie, einen Report mit Eingabewerten für den Programmlauf auf einem Selektionsbildschirm zu versorgen. Sie erarbeiten sich einfache und komplexe Selektionen und Selektionstexte und gestalten den Selektionsbildschirm völlig frei. Außerdem lernen Sie, was Textsymbole, Nachrichten und Varianten bedeuten, insbesondere im Zusammenhang mit dem Selektionsbildschirm.

Kapitel 12, »Interne Tabellen«, zeigt interne Tabellen, deren Bedeutung und verschiedene Formen. Sie erfahren, wie Sie eine interne Tabelle definieren, befüllen und zeilenweise verarbeiten. Zum besseren Verständnis wird auch in diesem Arbeitsschritt besonderer Wert auf die Ablaufverfolgung im ABAP Debugger gelegt.

In **Kapitel 13**, »Modularisierung von Programmen«, erfahren Sie, was Quelltextmodule, interne und externe Unterprogramme sowie Funktionsbausteine sind. Bei dieser Gelegenheit lernen Sie auch eine Möglichkeit für den Datenexport aus dem SAP-System kennen. Ein weiterer wichtiger Punkt dieses Kapitels ist die Übergabe von Daten und Datenstrukturen an gerufene Module.

In **Kapitel 14**, »ABAP in Eclipse«, beschäftigen wir uns mit der neuen, Eclipse-basierten ABAP-Entwicklungsumgebung. Wir zeigen Ihnen Download und Installation der Software, die Verbindung mit einem ABAP-System und machen dann die ersten Schritte in der neuen Umgebung.

Kapitel 15, »Core Data Services zur Abbildung von Datenmodellen«, führt Sie in die Welt der CDS Views und macht Sie mit dem vielleicht wichtigsten neuen Entwicklungswerk-

zeug vertraut, das in den letzten Jahren Einzug in die SAP-Welt gehalten hat. Sie lernen, wie Sie CDS Views anlegen und mithilfe von Assoziationen mit anderen Views und Tabellen verknüpfen. Sie erfahren, wie Sie mithilfe von Annotationen Ihren CDS View um Metadaten anreichern, die von zahlreichen Frameworks genutzt werden, um Ihnen das Entwicklerleben zu erleichtern.

Kapitel 16, »Weiterführende Themen«, gibt einen Ausblick und Vorgeschmack auf die zahlreichen Neuerungen, die in der SAP-Welt Einzug gehalten haben. Auch wenn ein Einsteigerbuch fortgeschrittene Themen wie Web Dynpro, OData-Services und cloudbasierte Entwicklung nicht in der Tiefe behandeln kann, kann es doch einen groben Überblick vermitteln und die wichtigsten Themen auf Ihrer geistigen SAP-Landkarte verorten.

Download der Codebeispiele

Um Ihnen zeitraubende Tipparbeit zu ersparen, können Sie die Quellcodes zu allen Beispiel-Listings von der Seite zu diesem Buch auf der Website von SAP PRESS herunterladen (*http://www.sap-press.de/5652*).

Danksagung

Es ist mir eine Freude, mich bei meiner Frau und meinen Kindern zu bedanken: Sie haben es mir durch viel Verzicht auf unser Familienleben und durch häusliche »Entlastungstätigkeiten« ermöglicht, dieses Buch fortzuschreiben.

Renate Fäcke-Kühnhauser danke ich, dass ich das Werk ihres leider verstorbenen Gatten weiterführen darf. Karl-Heinz Kühnhauser ist der ursprüngliche Autor dieses Buches, dem ich mich zutiefst verpflichtet fühle. Ich habe mich bemüht, alle Überarbeitungen so gut wie möglich in seinem Sinn vorzunehmen.

Als selbstständiger SAP-Berater danke ich meinen Kunden für spannende Projekte und Workshops, die mich das Thema ABAP immer wieder in neuem Licht sehen lassen.

Allen Lesern wünsche ich viel Spaß und viel Erfolg beim Lesen und Lernen!

Thorsten Franz
Freiberuflicher SAP-Berater, *operatics.de*, Bonn

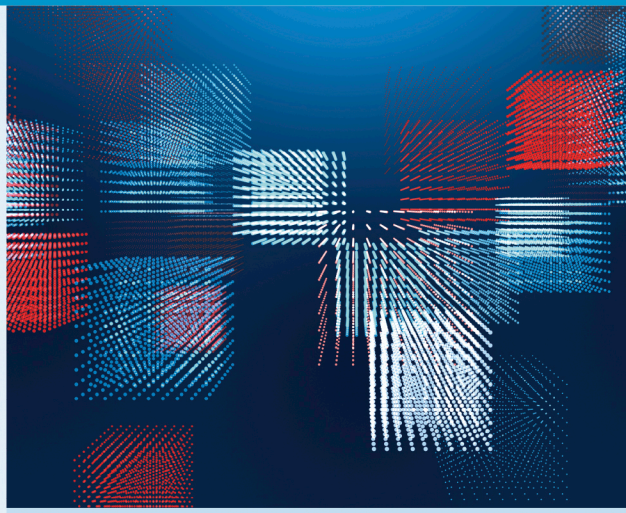
Inhalt

ABAP lernen leicht gemacht!

Mit diesem Standardwerk feiern Sie im Handumdrehen erste Erfolge mit selbst geschriebenem Quellcode. Es behandelt alle relevanten ABAP-Sprachelemente in kompakten Lerneinheiten – vom einzeiligen Report bis hin zur Programmablaufsteuerung und Datenübernahme. Steigen Sie mit CDS Views noch tiefer in die Welt der ABAP-Programmierung ein. Kommentierte Codebeispiele, viele Screenshots sowie wertvolle Tipps und Tricks sind dabei der Garant für Ihre Fortschritte.

Mit diesem Buch lernen Sie:

- Objekte im ABAP Dictionary anlegen
- Mit dem ABAP Editor und Eclipse arbeiten
- Reports erstellen
- Listen und Selektionsbildschirme ausgeben
- Berechnungen durchführen
- Mit verschiedenen Datentypen arbeiten
- Mit Zeichenketten umgehen
- Datenbanktabellen bearbeiten
- Schleifen und logische Ausdrücke verwenden
- Interne Tabellen verarbeiten
- Programme modularisieren
- CDS Views erstellen



Das Autorenteam

Thorsten Franz ist Gründer des SAP-Beratungsunternehmens operatics und seit mehr als 20 Jahren als Architekt, Berater, Entwickler, Projektleiter und Coach tätig. Karl-Heinz Kühnhauser leitete mehr als 20 Jahre ABAP-Schulungen in verschiedenen Branchen.

Programmierbeispiele stehen zum Download bereit

Rheinwerk Publishing

Gedruckt in Deutschland
Papier aus nachhaltiger Waldwirtschaft
Mineralölfreie Druckfarben

€ 39,90 [D] € 41,10 [A]
ISBN 978-3-8362-9354-9