HANSER

Vorwort

zu

„Oracle Application Express in der Praxis“

von Ralf Beckmann

ISBN (Buch): 978-3-446-43896-5
ISBN (E-Book): 978-3-446-43913-9

Weitere Informationen und Bestellungen unter
http://www.hanser-fachbuch.de/978-3-446-43896-5
sowie im Buchhandel

4 Wertelisten (List of Values)

■ *Wertelisten bieten Ihnen eine Möglichkeit, nur gewünschte Eingaben zuzulassen.*

Es gibt verschiedene Möglichkeiten, die Daten einer Anwendung konsistent zu halten. Sie können und sollten die Benutzereingaben prüfen. Oft ist es aber einfacher, einem Benutzer eines Systems erst gar nicht den Freiraum zu lassen, Daten nach Gutdünken zu erfassen.

APEX bietet Ihnen mit seinen *Wertelisten* ein Konstrukt, über welches Sie nur definierte Werte zur Erfassung anbieten können. Erfahren Sie in diesem Kapitel, wie Sie diese Möglichkeit in Ihren Programmen nutzen können.

In diesem Kapitel ...

- erfahren Sie, was *statische* und was *dynamische Wertelisten* sind.
- geht es um die Optimierung des Datenmodells.
- speichern Sie *Wertelisten* in den *Gemeinsamen Komponenten*.
- lernen Sie einen generativen Ansatz zur Erzeugung von *dynamischen Wertelisten* kennen.
- finden Sie heraus, wie Sie *Quick Picks* einsetzen können.

4.1 Was sind Wertelisten?

Wertelisten sind Datenstrukturen, die Ihnen in APEX an sehr vielen Stellen über den Weg laufen werden. *Wertelisten* auch *List of Values* oder kurz *LOVs* genannt, sind Sammlungen von Daten, die Sie den Nutzern Ihrer Anwendungen über verschiedene Item-Typen anbieten können. Diese *Seitenelemente* erlauben es dann, aus diesen Daten ein oder mehrere Einträge auszuwählen. Beispielsweise könnten Sie in der *Auftragsverwaltung* einen Verantwortlichen aus einer Stammdatentabelle auswählen lassen.

Unterschieden werden *dynamische* und *statische LOVs*. Bei den *dynamischen Wertelisten* ergeben sich die Inhalte aus den Tabellen der Datenbank. Sie werden also über Select-Abfragen definiert. Die grundlegende Struktur so einer Anweisung sieht wie folgt aus.

```
select <Anzeige-Spalte> d, <Schlüssel-Spalte> r
  from <Tabelle>
```

APEX geht bei den *dynamischen Wertelisten* davon aus, dass die erste Spalte des Statements in der Anwendung angezeigt wird, während die Spalte r zurückgegeben und in der Datenbank gespeichert wird.

Bei den *statischen LOVs* werden die Werte anders als bei den *dynamischen Wertelisten* fest vorgegeben und von APEX verwaltet.

Lists of Values können als Basis für *Popup-Wertelisten (Popup-LOV)* dienen, welche Sie für die Auswahl eines vorgegebenen Werts aus einer längeren Liste verwenden können. Sind nur wenige Einträge in einer *LOV* vorhanden, können diese als *Kontrollkästchen* (*Radiogroup*), aber auch als einfache *Auswahlliste* verwendet werden.

Soll eine Mehrfachauswahl möglich sein, so stellt APEX hierzu ebenfalls verschiedene Item-Typen zur Verfügung. Eine Übersicht ist in der Tabelle 4.1 zu finden. Die ausgewählten Werte werden bei einer Mehrfachauswahl dann als durch Doppelpunkt getrennte Liste zurückgegeben.

ACHTUNG: Bei dieser Art der Speicherung von 1:n-Zuordnungen liegt Ihr Datenmodell nicht mehr in der Dritten Normalform vor. Dies sollten Sie auf jeden Fall im Hinterkopf behalten, wenn Sie Seitenelemente zur Mehrfachauswahl in Ihren Anwendungen einsetzen.

Es ist natürlich möglich, z. B. über eigene Prozesse oder entsprechende Trigger in der Datenbank, aus dieser Notationsform die Daten in eine zweite in Relation stehende Tabelle zu transportieren. In dieser Tabelle wird für jede Auswahl ein eigener Datensatz erzeugt. Dieser besitzt dann als Fremdschlüssel die ID der Ausgangstabelle und den jeweiligen Schlüssel der Werteliste.

Beispiel:

Einer Aufgabe mit der ID 23 werden über eine Werteliste verschiedene Mitwirkende zugeordnet. Die Mehrfachauswahl liefert folgende Werte 1:2:4.

In einem normalisierten Modell würden daraus folgende Datensätze in der Mapping-Tabelle erzeugt.

ID , ID-Wertelistenelement

23 , 1

23 , 2

23 , 4

Im Einzelnen können Sie diese *Seitenelemente* wie in Tabelle 4.1 aufgeführt nutzen:

Tabelle 4.1 LOV-bezogene Seitenelemente

Item-Typ	Einsatzgebiet
Auswahlliste (Select List)	Eingabefeld mit der Möglichkeit, eine Werteliste aufzuklappen. Eine *Auswahlliste* bietet sich vor allem dann an, wenn es um eine begrenzte Zahl an Auswahlmöglichkeiten geht.
Kontrollkästchen (Check Box)	Auflistung verschiedener Werte, die durch Anhaken ausgewählt werden können. Eine *Check Box* ist in der Regel bei einer überschaubaren Anzahl von Einträgen sinnvoll.
Optionsgruppe (Radio Group)	Auflistung zum Anhaken genau eines Elements. *Optionsgruppen* sind für Listen mit wenigen Datensätzen brauchbar.
Popup-Werteliste (Popup List of Values)	Eingabefeld mit einem Icon, über das in einem separaten Fenster eine *Werteliste* eines Eintrags geöffnet werden kann. Dieses Item bietet eine Volltextsuche. Eine *Popup-Werteliste* sollten Sie verwenden, wenn die *Werteliste* viele Daten enthält.
Shuttle	Auswahlsteuerelement mit zwei Listen. Der Anwender kann Elemente aus der Vorgabeliste in die zweite Liste verschieben. Hierdurch ist eine Mehrfachauswahl möglich. Ein *Shuttle-Item* können Sie für eine überschaubare Datenmenge verwenden.
Listen-Manager (List Manager)	Ermöglicht die Auswahl mehrerer Einträge aus einer *Werteliste*. Die gewählten Einträge werden aufgelistet und können auch wieder entfernt werden. Ein *Listen-Manager* bietet Ihnen eine Popup-Werteliste, über die Sie aus einer großen Anzahl an Daten eine Auswahl treffen können. Dieser Datensatz kann dann einer Sammlung von Werten zugefügt werden.

Die Item-Typen legen Sie für alle diese *Seitenelemente* (Bild 4.1) über die Eigenschaft **Anzeigen als** fest.

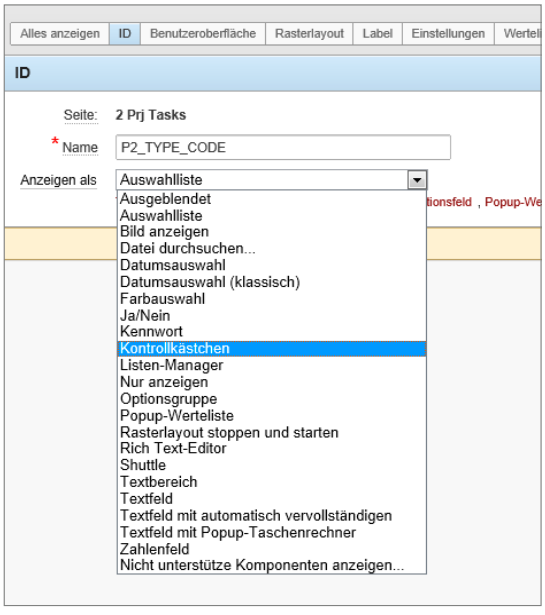

Bild 4.1
Item-Typen festlegen

Die eigentliche Definition der *Werteliste* erfolgt bei diesen *Seitenelementen* über deren Eigenschaften-Region **Werteliste**.

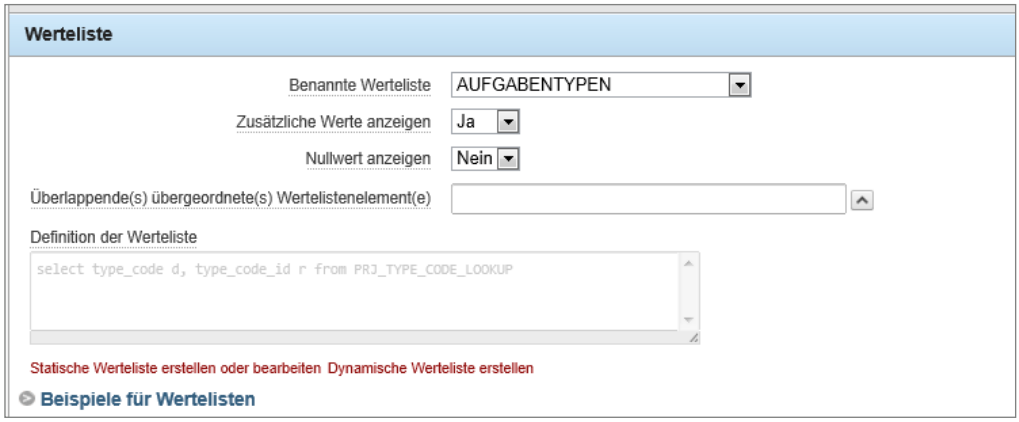

Bild 4.2 Definition einer Werteliste

Über diese Region haben Sie verschiedene Möglichkeiten, auf die *Werteliste* Einfluss zu nehmen. Sie können eine *Werteliste* direkt als Select-Statement definieren. *Wertelisten,* die Sie an verschiedenen Stellen Ihrer Anwendung einsetzen, können Sie speichern und an dieser Stelle als sogenannte **Benannte Werteliste** dem *Seitenelement* zuordnen.

Wie genau Sie *Wertelisten* in Ihren Anwendungen benutzen können und welche Implementierungsvarianten Ihnen zur Verfügung stehen, zeige ich Ihnen in den folgenden Abschnitten.

4.2 Statische Wertelisten

Ein Ziel beim Einsatz von *Wertelisten* ist es, den Benutzer bei der Eingabe seiner Daten enger zu führen und damit Fehleingaben sowie unterschiedliche Schreibweisen zu verhindern. Schauen Sie sich die Seite 2 noch einmal an, so findet sich ausgehend von dieser Überlegung recht schnell ein Kandidat für den Einsatz einer *Werteliste*. Bisher musste der Status einer Aufgabe umständlich eingetippt werden. Fehleingaben werden dabei durch eine entsprechende *Validierung* vermieden.

AUFGABE: Erstellen Sie eine Werteliste, die als Basis für die Eingabe des Status-Felds verwendet werden kann, und ändern Sie den Item-Typ für das Statusfeld auf eine *Optionsgruppe*.

In Abschnitt 3.5 haben Sie für das *Seitenelement* Status eine *Validierung* erstellt, die folgende Werte als Eingaben akzeptiert.

- Offen
- In Bearbeitung
- Fertig

Für solch einen Fall können Sie bei APEX auf *statische Wertelisten* zurückgreifen.

Bei einer *statischen Werteliste* legen Sie als Entwickler die möglichen Werte fest. Die Einträge in einer *statischen Werteliste* können nicht durch die Anwender verändert werden. Allerdings ist bei dieser Art *Werteliste* keine Stammdatentabelle mit entsprechender Bearbeitungsseite notwendig. Für einfache Listen, deren Werte sich nicht oder nur sehr selten ändern, stellt diese Form eine schlanke Alternative zu den *dynamischen LOVs* dar.

Wie einleitend in diesem Kapitel bereits dargestellt, definieren Sie eine *Werteliste* über die Eigenschaften des entsprechenden Items.

In der Region **Werteliste** des Items P2_PROCESSING_STATUS finden Sie unterhalb des Eingabefelds **Definition der Werteliste** einen Link namens **Statische Werteliste erstellen oder bearbeiten**. Betätigen Sie diesen, öffnet sich die in Bild 4.3 gezeigte Seite.

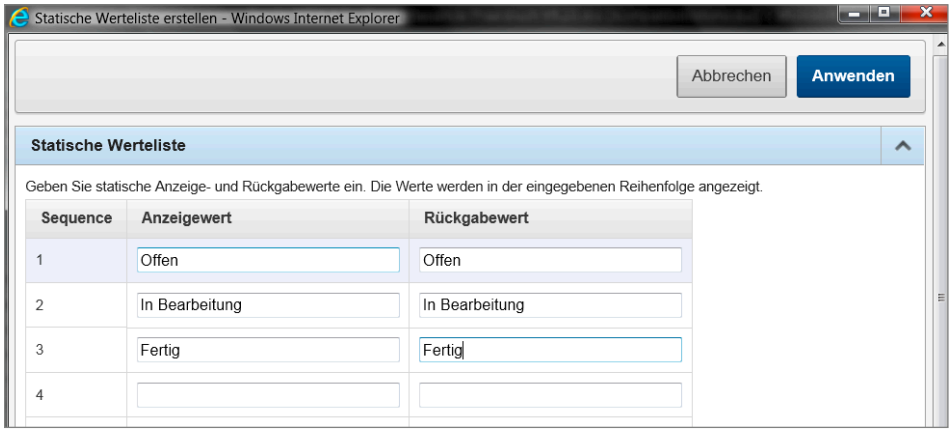

Bild 4.3 Definition einer statischen Werteliste

Hier geben Sie die verschiedenen Daten ein, die über dieses Seitenelement ausgewählt werden können. Der **Anzeigewert** ist der Eintrag, der später in der Seite zu sehen sein wird, während der **Rückgabewert** als Ergebnis des Items zurückgegeben wird.

Nachdem Sie den Button **Anwenden** betätigt haben, sehen Sie folgende Einträge (Bild 4.4) in dem Feld **Definition der Werteliste**.

Diese Aufzählung listet alle möglichen Einträge Ihrer *statischen LOV* auf.

Die Frage, ob Nullwerte angezeigt werden sollen, beantworten Sie mit **Nein**.

Damit beim Anlegen einer neuen Aufgabe der korrekte Status gesetzt ist, geben Sie in der Region **Standardwert** als **Statischen Text** den Wert *Offen* ein. Speichern Sie dann die Änderungen.

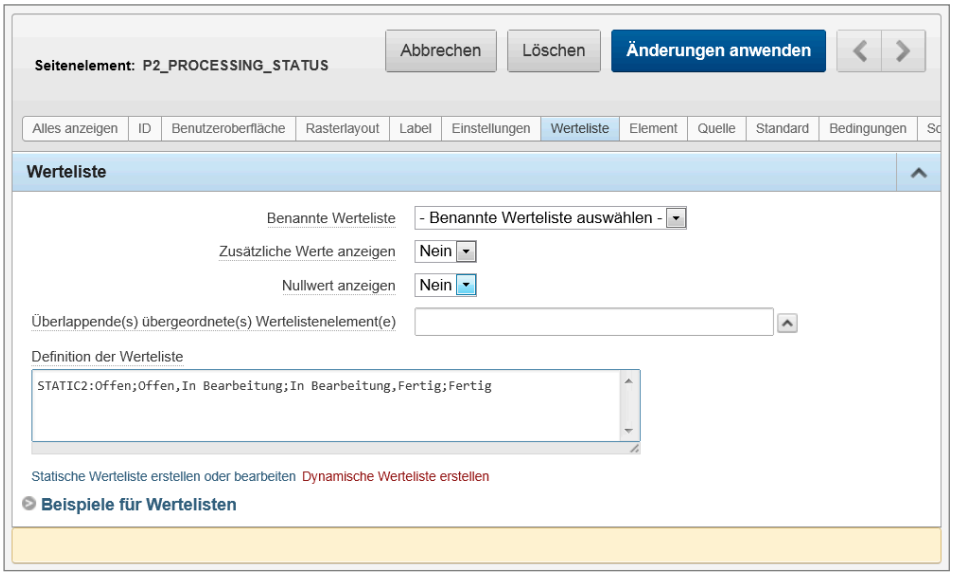

Bild 4.4 Verwendung einer gespeicherten Werteliste

Jetzt können die Anwender der *Aufgabenverwaltung* nur noch vorgegebene Werte auswählen. Durch die Option **Nullwerte anzeigen: Nein** erreichen Sie darüber hinaus, dass immer ein Wert erfasst ist. Aus diesem Grund wird die *Validierung* **P2_PROCESSING_STATUS check**, die Sie in Abschnitt 3.5.3 erstellt haben, nicht mehr benötigt und kann gelöscht werden. Sie erledigen das am schnellsten, indem Sie über die *Seitendefinition* der Seite 2 die *Validierung* öffnen und dort auf die Schaltfläche **Löschen** klicken.

Das Ergebnis der Anpassungen zeigt Bild 4.5.

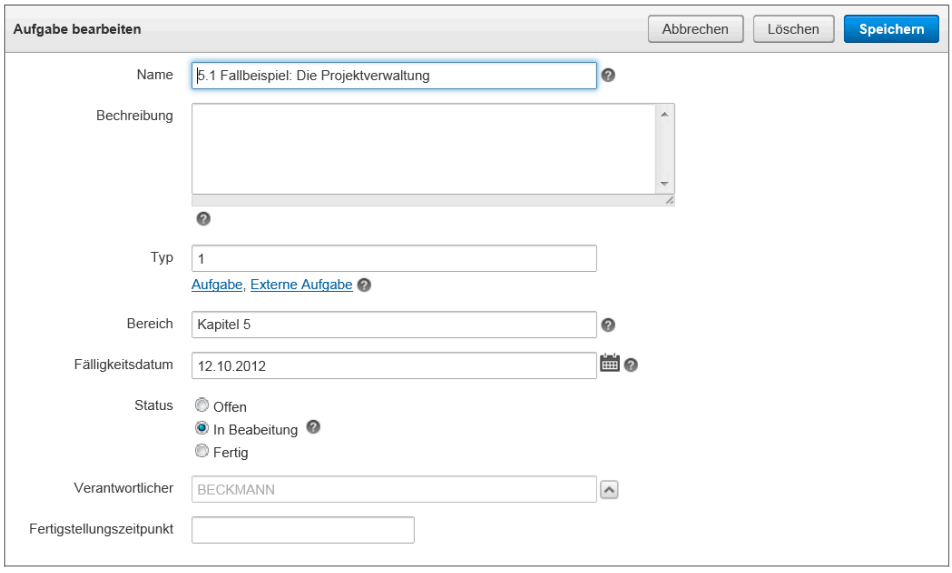

Bild 4.5 Statuseingabe als Radio Group

4.3 Selbstlernende dynamische Wertelisten

Der Einsatz einer *dynamischen Werteliste* ist im Vergleich zur statischen Variante mit etwas mehr Aufwand verbunden. Bei dieser Art LOVs werden die Daten aus der Datenbank gelesen.

Bei den *Seitenelementen* **Typ** (P2_TYPE_CODE) und **Bereich** (P2_CLASSIFICATION) können Eingabefehler vermieden werden, wenn Sie selbstlernende Listen verwenden.

AUFGABE: Konfigurieren Sie die Items P2_TYPE_CODE und P2_CLASSIFICATION jeweils zu einer selbstlernenden Liste um, so dass Abweichungen bei der Wahl der Begrifflichkeiten verringert werden.

Öffnen Sie dazu die Eigenschaften des Items **P2_TYPE_CODE**.

Wählen Sie für dieses *Seitenelement* den Typ **Textfeld automatisch vervollständigen** und setzen Sie anschließend die **FORM-ELEMENTBREITE** sowie die **Max Breite** des Felds auf 60 Zeichen.

Nun müssen Sie über die Region **Werteliste (LOV)** definieren, welche Werte das Autocomplete-Item annehmen kann. Dazu tragen Sie das SQL Select Statement `select distinct type_code from prj_tasks` in das Feld **Definition der Werteliste** ein. Dieses Statement liest alle bereits in dieser Spalte getätigten Eingaben aus der Tabelle *prj_tasks*. Mit dem Befehl `distinct` werden dabei die doppelten Einträge beseitigt.

Schließen Sie den Vorgang wieder über **Änderungen Anwenden** ab.

Anschließend verfahren Sie für das Item P2_CLASSIFICATION **genauso wie für P2_TYPE_CODE.** Die notwendige Select-Anweisung sieht für dieses Feld wie folgt aus:

```
select distinct classification from prj_tasks.
```

Passen Sie nun noch für alle anderen Items die **Form-Elementbreite** des Felds auf 60 Zeichen an. Filtern Sie hierzu nach dem Öffnen des ersten Items auf die Region **Angezeigt** und steppen Sie wie in Abschnitt 3.1 beschrieben mit dem kleinen Pfeil neben dem **Änderungen anwenden**-Button durch die Items.

Schließen Sie die letzte Anpassung über **Änderungen anwenden** ab und starten Sie die Anwendung (Button **Ausführen**). Melden Sie sich gegebenenfalls an der Aufgabenverwaltung neu an.

Das Ergebnis der bisherigen Bemühungen stellt sich wie in Bild 4.6 auf der nächsten Seite dar.

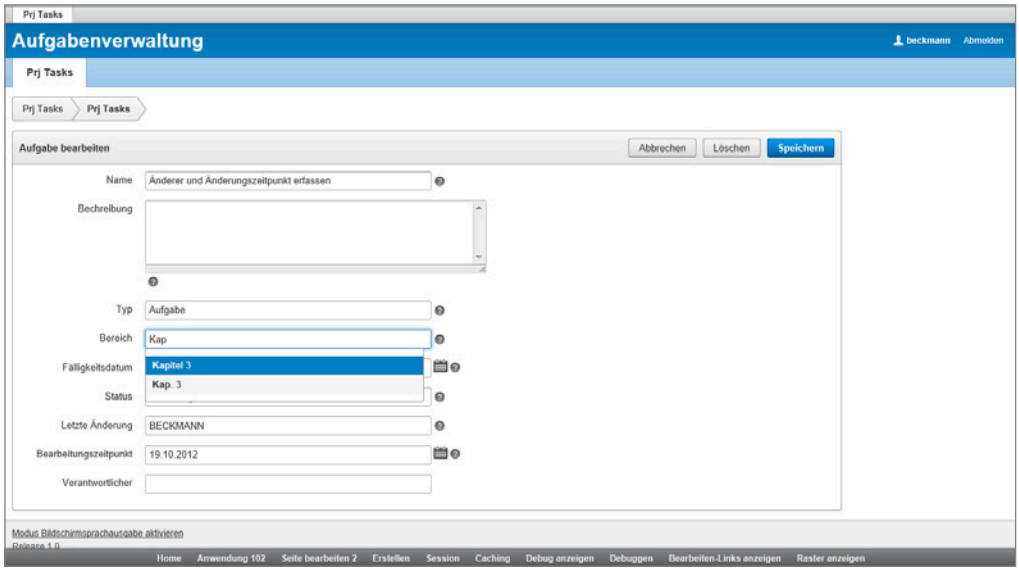

Bild 4.6 Verbesserte Seite 2 – „Aufgaben bearbeiten“

4.4 Dynamische Wertelisten auf Basis einer Lookup-Tabelle

Das folgende Beispiel ist etwas komplexer. Ziel dieses Abschnitts ist es, die *Aufgabenverwaltung* um eine Stammdatentabelle zu erweitern. Diese soll die möglichen Aufgabentypen beinhalten, die in Seite (2)-Aufgabe bearbeiten genutzt werden können. Die Auswahl der Aufgabentypen soll dann über eine *Auswahlliste* erfolgen.

Das Beispiel führt nicht nur in das Thema *dynamische Wertelisten* ein, sondern soll Ihnen auch verdeutlichen, welche Änderungen in einem produktiven System notwendig sind, wenn das Datenmodell einer Anwendung im Nachhinein erweitert wird.

Folgende Punkte werden in diesem Abschnitt betrachtet.

1. Normalisieren des Datenmodells (Erstellung einer *Lookup-Tabelle*)
2. Anpassung bestehender Elemente der Anwendung
3. Erstellung von Seiten zur Verwaltung der Stammdaten
4. Datenquelle eines Seitenelements anpassen
5. Erstellen der dynamischen Werteliste

4.4.1 Lookup-Tabelle mit dem Objektbrowser erstellen

Führen Sie sich den Aufbau der Tabelle *prj_tasks* noch einmal vor Augen und betrachten Sie das *Seitenelement* **Aufgabentypen** etwas genauer. Diesem Item liegt die Spalte *type_code* aus der Tabelle *prj_tasks* zugrunde. Bisher wurden in den verwendeten Beispielen die Einträge „Aufgaben“ und „Meilensteine“ dort erfasst. Die Nutzer der *Aufgabenverwaltung* wurden durch eine *Validierung* gezwungen, einen Aufgabentyp einzugeben. Die konkrete Schreibweise wurde allerdings nicht vorgegeben.

In solchen Fällen bietet es sich an, die möglichen Werte in einer Stammdatentabelle zu speichern. Für solche Tabellen können Sie mit APEX sehr schnell Seiten zur Verwaltung der Daten zur Verfügung stellen. Für den Inhalt sind dann später die Anwender zuständig.

AUFGABE: Normalisieren Sie das Datenmodell und erweitern Sie das System um eine Stammdatentabelle, die die möglichen Aufgabentypen aufnimmt.

Technisch ist dies auf den ersten Blick kein Problem. Sie erstellen eine entsprechende Tabelle und ersetzen in der Tabelle *prj_tasks* die Spalte `type_code` durch einen *Fremdschlüssel* – also einen Verweis auf die Stammdatentabelle.

Auf den zweiten Blick sieht das dann aber so aus.

In unserem Fallbeispiel wurde ja bereits mit der *Aufgabenverwaltung* gearbeitet. In der Tabelle *prj_tasks* befinden sich somit Daten.

Es gibt verschiedene Wege, Änderungen an einem produktiven System durchzuführen. Technisch ist Folgendes dazu notwendig.

Sie müssen die bereits verwendeten Aufgabentypen in die neue Stammdatentabelle übernehmen. Dabei sollten Sie einen *künstlichen Schlüssel* für jeden Aufgabentyp erzeugen. Anschließend ist noch die Spalte `type_code` durch die neue Fremdschlüsselspalte zu ersetzen.

Sie sehen bereits an diesem einfach anmutenden Beispiel, wie aufwendig es sein kann, ein produktives System anzupassen. Das gilt vor allem bei strukturellen Änderungen des Datenmodells.

Die geschilderte Aufgabe lässt sich mit entsprechenden Datenbanktools und einigen SQL-Befehlen bewerkstelligen. APEX bietet für diesen Fall, zugegeben etwas versteckt, eine Funktion, welche Ihnen diese Arbeit deutlich erleichtert.

Begeben Sie sich in der Entwicklungsumgebung in das Modul **SQL Workshop** und öffnen Sie dort den **Objektbrowser.** Im *Objektbrowser* markieren Sie die Tabelle *prj_tasks*.

Wählen Sie – wie in Bild 4.7 gezeigt – das Register **Tabelle** und dann den Eintrag **Lookup-Tabelle erstellen**. Es startet ein Assistent, welcher Ihnen beim Auslagern der Spalte type_code in eine separate Tabelle behilflich sein wird.

Im ersten Schritt werden Sie gebeten, die Spalte auszuwählen, für die Sie eine *Lookup-Tabelle* erstellen möchten. Markieren Sie den Spaltennamen **Type_Code** und klicken Sie auf **Weiter >**.