

Ewald Brochhausen, Jürgen Kielisch,
Jürgen Schnerring, Jens Staeck

mySAP® HR – Technische Grundlagen und Programmierung



Inhalt

Vorwort	11
1 Einleitung	13
1.1 Dimensionen von mySAP HR	13
1.2 Die Möglichkeiten der Anpassung	14
1.3 Der Aufbau dieses Buches	16
1.4 Weiterführende Informationen und Kontakt	19
2 Datenstrukturen im HR	21
2.1 Datenstrukturen im Kontext	21
2.2 Stammdaten der Personaladministration	22
2.2.1 Infotypen der Personaladministration	22
2.2.2 Unterteilung der Infotypen – der Subtyp	24
2.2.3 Die Objektidentifikation	26
2.2.4 Zeit und Zeitbindung der Infotypen	26
2.2.5 Einzelbild und Listbild	28
2.2.6 Vorschlagswerte für Infotypen	29
2.2.7 Der Header eines Infotyps	30
2.2.8 Merkmale und Bildmodifikatoren	32
2.2.9 Infotypen den Ländern zuordnen	35
2.2.10 Technische Datenstruktur der Infotypen	36
2.2.11 Datenstrukturen und Tabellen: Infotyp der Personaladministration	44
2.2.12 Infotyp-Views	44
2.3 Daten des Organisationsmanagements und der Personalplanung	45
2.3.1 Datenmodell	46
2.3.2 Infotypen der Personalplanung	51
2.3.3 Technische Datenstruktur der Infotypen	52
2.3.4 Tabelleninfotypen	57
2.3.5 Externe Objekttypen	59
2.3.6 Externe Infotypen	60
2.3.7 Datenstrukturen und Tabellen: Infotyp der Personalplanung	61
2.3.8 Konsistenzüberprüfung des Datenmodells	61
2.3.9 Konsistenzüberprüfung der Infotypen	62
2.4 Zeitwirtschaftsdaten	63
2.4.1 Stammdaten der Zeitwirtschaft	64
2.4.2 Zeitereignisse	64

2.4.3	Input der Zeitauswertung	66
2.4.4	Ergebnisse der Zeitauswertung	68
2.5	Abrechnungsdaten	70
2.5.1	Zentrale Informationen zur Abrechnung	71
2.5.2	Ergebnisdaten von Abrechnungen	71
2.5.3	Cluster-Directory	77
2.5.4	Abrechnungsdaten für Auswertungen	78

3 Lesen und Bearbeiten von Daten 81

3.1	Die logischen Datenbanken im HR	81
3.1.1	Logische Datenbank PNP für die Personalstammdaten	83
3.1.2	Logische Datenbank PNPCE für Personalstammdaten	96
3.1.3	Logische Datenbank PCH für die Personalplanung	98
3.2	Zugriffe ohne logische Datenbank	104
3.3	Die Nutzung von Makros	105
3.3.1	Überblick	105
3.3.2	Makros in der logischen Datenbank PNP	107
3.3.3	Makros in der logischen Datenbank PCH	113
3.4	Funktionsbausteine	114
3.4.1	Eigenschaften	114
3.4.2	Verwendung von Funktionsbausteinen im HR	115
3.4.3	Nutzung von Merkmalen	118
3.5	Zugriff auf Cluster	123
3.5.1	Allgemeine Vorgehensweise	123
3.5.2	Abrechnungsergebnisse	125
3.6	Erweiterungen mit Customer-Exits und Business Add-Ins	129
3.6.1	Customer-Exits	129
3.6.2	Business Add-Ins	131

4 Rollen und Berechtigungen 137

4.1	Das SAP-Berechtigungskonzept	137
4.1.1	Berechtigungsobjekte, Berechtigungen, Profile	137
4.1.2	Rollenkonzept, Profilgenerator	139
4.1.3	Funktionsweise des Profilgenerators	141
4.2	Berechtigungen im Kontext von SAP HR	143
4.2.1	Berechtigungsobjekte	143
4.2.2	Berechtigungslevel – abgestufte Schreibberechtigungen	153
4.2.3	Organisationsschlüssel	158
4.2.4	Strukturelle Berechtigungsprüfung	161
4.2.5	HR-Berechtigungs-hauptschalter	168
4.2.6	Zeitabhängigkeit	170
4.2.7	Implementierung	174
4.2.8	Erweiterungsmöglichkeiten	179
4.2.9	Zusammenspiel von Anwendung und Berechtigungsprüfung	189
4.2.10	Fehlersuche	191

5 Anpassungen in den Applikationen 193

5.1	Personaladministration	193
5.1.1	Komponenten im Repository	193
5.1.2	Erweiterung von Infotypen	195
5.1.3	Anlegen von Infotypen	206
5.1.4	Erweiterung von Infotypen für die Schnellerfassung	216
5.1.5	Infotypübergreifende Erfassung mit der Maßnahmenschnellerfassung	221
5.2	Organisationsmanagement	225
5.2.1	Erweiterung von Infotypen	225
5.2.2	Anlegen von Infotypen	230
5.3	Zeiterfassung	238
5.4	Abrechnung und Zeitauswertung	241
5.4.1	Steuerung der Abrechnung und Zeitauswertung	241
5.4.2	Funktionen	244
5.4.3	Operationen	247

6 Reporting im HR 251

6.1	Stammdaten	251
6.1.1	Aufbau eines Reports für Infotypen der Personaladministration	251
6.1.2	Aufbau eines Reports für Infotypen der Zeitwirtschaft	264
6.2	Organisationsmanagement	268
6.2.1	Sequenzielle Auswertungen	268
6.2.2	Strukturelle Auswertung	269
6.3	Abrechnungsdaten	272
6.4	Zeitwirtschaft	277
6.5	Aufbereitung der Ausgabe mit dem ABAP List Viewer	280

7 Reportingwerkzeuge 287

7.1	Auswertungsmöglichkeiten im Überblick	287
7.2	Berichte im Menü	289
7.3	Verwendung von HIS	291
7.4	SAP Query	296
7.4.1	Arbeitsbereiche	296
7.4.2	Erstellen einer Query	296
7.4.3	Anlegen von InfoSets	305
7.4.4	Zuordnung zu Benutzergruppen	308
7.4.5	Lokale Felder in Queries	309
7.4.6	Anlegen von Zusatzfeldern im InfoSet	313

7.4.7	Anlegen von InfoSet-übergreifenden Zusatzfeldern	317
7.4.8	Definition von Schaltern	323
7.4.9	Abrechnungsinfortypen	333
7.5	Ad-hoc-Query	339

8 Erstellung von Formularen mit dem HR-Formular-Workplace 347

8.1	Die Möglichkeiten der Erstellung	347
8.2	Erstellung von Formularen	348
8.3	Anlegen der Metadaten mit dem HR-Metadaten-Workplace	349
8.4	HR-Formular-Workplace	357
8.5	Grafische Ausgabe	360

9 Werkzeuge für Schnittstellen 367

9.1	Programmierung mit BAPIs	367
9.1.1	Business-Objekte und BAPIs	367
9.1.2	HR-Objekte im BOR	369
9.1.3	Verwendung von BAPIs	371
9.2	Interface Toolbox	375
9.2.1	Export der Daten	376
9.2.2	Konfiguration des Interface-Formats	379
9.2.3	Konfiguration des File-Layouts	381

10 Employee Self-Service 387

10.1	Funktionsumfang	387
10.1.1	Betriebswirtschaftliche Sicht	387
10.1.2	Navigation	388
10.2	Die Rolle »Employee«	389
10.2.1	Rollenkonzept	389
10.2.2	Einzel-, Länder- und Sammelrollen	390
10.3	Übersicht Internet Transaction Server	392
10.3.1	Installationsvarianten	392
10.3.2	Die ITS-Dateien	394
10.3.3	Übersicht über die wichtigsten ITS-Serviceparameter	398
10.3.4	Arbeitsweise des ITS	401
10.3.5	HTML-Business-Funktionen	403
10.4	Programmiermodelle	405
10.4.1	Internet-Services basierend auf HTML-Templates (IAC)	407
10.4.2	FlowLogic	410
10.4.3	SAP GUI für HTML	414

10.5	Design- und Funktionserweiterungen	415
10.5.1	Die User-Exits des ESS	416
10.5.2	Designanpassungen	418
10.5.3	Felder oder Ablauflogik ändern	423
10.5.4	Einen neuen länderabhängigen Service erstellen	425
10.5.5	Einen neuen Service erstellen	428
10.6	Web-enabling von Reports	430
10.6.1	ESS-Report-Framework	430
10.6.2	Beispiel-Anbindung eines Reports in ESS	433
10.7	Life-and-Work-Events	435
10.7.1	Konzept der Life-and-Work-Events	435
10.7.2	Voraussetzungen und Funktionalität	438
10.7.3	Das Framework	440
10.7.4	L&W-Tabellen im Überblick	449
10.7.5	Customizing	450
10.7.6	Erstellung eines neuen L&W-Events	456

A	ESS-Szenarien	459
----------	----------------------	------------

B	Literaturempfehlungen	463
----------	------------------------------	------------

	Die Autoren	465
--	--------------------	------------

	Index	467
--	--------------	------------

Vorwort

Seit der ersten Installation eines R/3 HR sind mittlerweile mehr als dreizehn Jahre vergangen. Während dieser Zeit wurde vieles weiterentwickelt und viel Funktionalität ergänzt. Ständig steigende Anforderungen an eine Human-Resources-Software führten zu Erweiterungen des Standards in Form von Customizing-Möglichkeiten und neuen Werkzeugen. Waren am Anfang oft noch Modifikationen des Systems erforderlich, um die spezifischen Besonderheiten der Unternehmen zu berücksichtigen, wurden im Lauf der Entwicklung immer mehr Möglichkeiten geschaffen, Kundenanpassungen gegenüber Änderungen im Standard zu schützen. Beispiele hierfür sind die Möglichkeiten, die Datenbasis beliebig zu ergänzen, die Standardbilder zu erweitern oder ABAP-Coding modifikationsfrei zu ersetzen oder hinzuzufügen.

Mit der Ausweitung der Anpassungsmöglichkeiten wurde es manchmal auch schwieriger, zu entscheiden, welche Vorgehensweise im konkreten Fall aus den verschiedenen Alternativen zu wählen ist. Um hierbei Hilfestellung zu geben, haben wir – auch auf Anregung vieler Anwender des SAP HR hin – die wichtigsten Grundlagen des HR in diesem Buch zusammengefasst. Hierbei geht es vor allem um die Besonderheiten des HR, weniger um allgemeine anwendungsübergreifende Grundlagen des SAP-Systems. Für die Nutzung dieses Buches sollten daher bereits grundlegende Kenntnisse der ABAP-Programmierung und von SAP HR vorhanden sein. Hierzu sei auf die im Anhang erwähnten Bücher verwiesen.

Wir freuen uns, dass der Verlag nach zwei Jahren die Neuauflage dieses Buches beschlossen hat. Der Erfolg der ersten Auflage gibt uns in unserem Bemühen Recht, zu einem spezifischen Segment der SAP-Software ein eigenes Programmier- und Technologie-Buch zu schreiben.

Es bleibt uns noch die angenehme Aufgabe des Dankens. Unser Vorhaben wurde von einer Vielzahl von Freunden und Kollegen unterstützt. Ihnen allen sei für ihren wertvollen Rat und ihre Hilfe gedankt. In diesem Zusammenhang seien Ulf Bangert, Klaus Billig, Bärbel Bohr, Udo Klein, Heiko Schultze und Matthias Wengner besonders erwähnt. Auch dem Verlag Galileo Press sei gedankt, hier insbesondere Florian Zimniak. Die

professionelle Unterstützung in Fragen des Layouts und in den verlagstechnischen Angelegenheiten hat zum Erscheinen dieses Buches wesentlich beigetragen.

Walldorf, im März 2005

**Ewald Brochhausen – Jürgen Kielisch –
Jürgen Schnerring – Jens Staack**

6 Reporting im HR

In diesem Kapitel wird dargestellt, wie Reports im HR für die verschiedenen Applikationen aussehen können. An Beispielen wird gezeigt, welche grundlegenden Elemente bei der Erstellung der Reports im HR verwendet werden.

6.1 Stammdaten

Die Infotypen der Personaladministration sind in dem Bereich von 0000 bis 0999, die Infotypen der Zeitwirtschaft in dem Bereich von 2000 bis 2999 zu finden. Hinzu kommen eventuell kundeneigene Infotypen im Bereich ab 9000. Wie bei einer Auswertung der Stammdaten sinnvoll vorzugehen ist, wird an den nächsten Beispielen auf der Basis der logischen Datenbank PNP erläutert.

6.1.1 Aufbau eines Reports für Infotypen der Personaladministration

Erstellen einer einfachen Liste

In dem dargestellte Beispiel sollen die bereits in Kapitel 3 erläuterten Verfahren um weitere Hinweise bei der Erstellung von Reports ergänzt werden. Als erstes Beispiel soll eine einfache Liste erstellt werden, in der zum Monatsende die Bankverbindung der Mitarbeiter mit Kontonummer, Bankleitzahl und Name des Geldinstituts aufgeführt ist (siehe Abbildung 6.1).

Liste Bankverbindungen

Für das Selektionsbild muss die geeignete Reportklasse verwendet werden. Sinnvoll ist hier die Stichtagsbetrachtung mit der Standard-Reportklasse `XX_10001`. Der Aufbau dieser Reportklassen wurde bereits in Abschnitt 3.1 erklärt. Ebenfalls bei den Eigenschaften des Reports wird die logische Datenbank `PNP` eingetragen. Hieraus ergibt sich das in Abbildung 6.2 dargestellte Selektionsbild.

Reportklasse

In der Listüberschrift soll das Stichtagsdatum in der Form `xx.xx.xxxx` erscheinen. Hierzu wird die Variable `<&0>` mit ergänzenden Punkten in der Gesamtlänge (10) der anzuzeigenden Variablen in die Listüberschrift bei den Textelementen aufgenommen (siehe Abbildung 6.3).

Listüberschrift

Bankverbindung der Mitarbeiter am 30.11.2002

Pernr	Name	Bankverbindung
00001006	Awad, Yasmin	Keine Bankverbindung
00001009	Braunstein, Herbert	564789123 10050033 Dresdner Bank Berlin
00001005	Gutjahr, Hanno	258963147 23984899 Vereinsbank München
00001001	Maier, Michaela	1258693 20050000 Deutsche Bank Hamburg
00001000	Müller, Anja	12345688 23984899 Vereinsbank München
00001008	Müller, Hilde	123456987 32902212 Raiffeisenbank Breitenrunn
00001004	Paulsen, Olaf	741852963 20050000 Deutsche Bank Hamburg
00001003	Pfändli, Stefan	258369741 12388823 Deutsche Bank Paderborn
00001010	Schmidtrohr, Frank	Keine Bankverbindung
00001007	Ulrich, Hanna	582369741 23022200 Volksbank Marschfelden
00001002	Zaucker, Ulrike	253571587 23984899 Vereinsbank München

Abbildung 6.1 Mitarbeiterliste mit Bankverbindung

weitere Selektionen Suchhilfen Sortierung

Stichtag

heute

anderer Stichtag

Stichtag 30.11.2002

Selektion

Personalnummer

Abbildung 6.2 Selektionsbild mit Stichtag

Textelemente Bearbeiten Springen Hilfsmittel Umfeld System Hilfe

ABAP Textelemente: Listüberschriften ändern Sprache Deutsch

Programm YTHR_DATE aktiv

Textsymbole Selektionstexte Listüberschriften

Listenüberschrift

.....1.....+.....2.....+.....3.....+.....4.....+.....5.....+.....6.....+.....7

Anspruchsjahre am &0.....

Abbildung 6.3 Textelemente zum Report »Anspruchsjahre«

Deklarationen Am Anfang des Reports erfolgen die notwendigen Deklarationen. Für die logische Datenbank ist die Anweisung `TABLES: pernr` erforderlich. Verarbeitet werden die Infotypen 0002 (*Daten zur Person*) und 0009 (*Bank-*

verbindung). Weiterhin werden die notwendigen Variablen `coname` und `bankname` definiert.

```
REPORT ythr_bank LINE-SIZE 132.  
TABLES: pernr.  
INFOTYPES: 0002, 0009.  
DATA: coname(40),           "Nachname, Vorname  
      bankname(40).        "Name der Bank
```

Die Ermittlung des Vorschlagswertes im Selektionsbild erfolgt zum Zeitpunkt `INITIALIZATION`. In dem Modul-Pool `sapfp500` befinden sich einige hilfreiche Routinen für das Berechnen von Zeitpunkten wie z.B. Addition einer Anzahl von Monaten zu einem gegebenen Datum. Hier wird mit dem externen Aufruf `last_day_in_month` aus dem Tagesdatum der Monatsletzte ermittelt. Dieser Wert wird als `pnpbegda` in dem Selektionsbild vorgeschlagen und kann bei Bedarf noch überschrieben werden.

Vorschlagswerte

```
INITIALIZATION.  
  PERFORM last_day_in_month(sapfp500)  
  USING sy-datum prop_date.  
  pnpbegda = prop_date.
```

Für die Zuweisung des Datums in der Listüberschrift wird zum Zeitpunkt `INITIALIZATION` das eventuell veränderte Datum in die Systemvariable `syst-tvar0` geschrieben.

```
START-OF-SELECTION.  
  WRITE pn-begda TO syst-tvar0 DD/MM/YYYY.
```

Zum Zeitpunkt `GET pernr` erfolgt die Verarbeitung für jede Personalnummer. Zunächst werden alle Werte initialisiert, die im Report berechnet werden. Die aktuell gültigen Werte der Infotypen 0002 und 0009 (mit dem Subtyp 0) werden in die Kopfzeilen der internen Tabellen `P0002` und `P0009` gestellt. Der Name wird aus Nachname und Vorname zusammengesetzt und der Name der Bank gelesen. Schließlich erfolgt die Ausgabe in eine Liste.

GET PERNR

```
GET pernr.  
  PERFORM clear_all.  
  rp_provide_from_last p0002 space pn-begda pn-begda.  
  rp_provide_from_last p0009 '0' pn-begda pn-begda.  
  PERFORM concat_name USING coname.  
  PERFORM read_bankname USING bankname.
```

```

WRITE: / pernr-pernr,
        coname,
        p0009-bankn(10),
        p0009-bankl(8),
        bankname.
END-OF-SELECTION.

```

Die einzelnen Routinen haben die im Folgenden beschriebenen Aufgaben. Die reportspezifischen Felder `coname` und `bankname` werden initialisiert, bei umfangreicheren Reports könnte dies auch in einem eigenen Include erfolgen.

```

FORM clear_all.
  CLEAR: coname, bankname.
ENDFORM.

```

Aufbereitung Name Aus den aktuell gültigen Feldern `p0002-nachn` und `p0002-vorna` wird der Name in der gewünschten Darstellung gebildet.

```

FORM concat_name USING p_conname.
  CONCATENATE p0002-nachn p0002-vorna
  INTO p_conname SEPARATED BY ' ', '.
ENDFORM.

```

Interface FI Der Name der Bank ist in Tabellen des FI gespeichert. Zum Lesen des Banknamens wählt man einen geeigneten Funktionsbaustein des HR-Interfaces. Diese beginnen mit »HRCA«. Für die Zwecke dieses Reports ist der Funktionsbaustein `HRCA_READ_BANK_ADDRESS_2` gut geeignet, der auf der Basis der Importparameter `bank_country` (Bankland) und `bank_number` (Bankleitzahl) eine Reihe Informationen der Bank in der Struktur `bankdata` zurückliefert. Benötigt wird hieraus das Feld `banka` (Name des Geldinstituts).

```

FORM read_bankname USING p_bankname.
  DATA : bankdata TYPE bnka_bf.
  CALL FUNCTION 'HRCA_READ_BANK_ADDRESS_2'
  EXPORTING
    bank_country = p0009-banks
    bank_number  = p0009-bankl
  IMPORTING
    bank_data    = bankdata
  EXCEPTIONS
    not_found    = 1

```

```

OTHERS          = 2.
IF sy-subrc = 0.
  p_bankname = bankdata-banka.
ELSE.
  p_bankname = 'Keine Bankverbindung'.
ENDIF.
ENDFORM.

```

Auswertung von Wiederholungsstrukturen

In einigen Infotypen der Stammdaten können die Ausprägungen von Eigenschaften mehrfach in tabellarischer Form erfasst werden. Beispiele hierfür sind die Erfassung von Lohnarten in den Infotypen 0008 (*Basisbezüge*) und 0052 (*Verdienstversicherung*) oder die in Abbildung 6.4 dargestellten Datumsfelder des Infotyps 0041 (*Datumsangaben*).

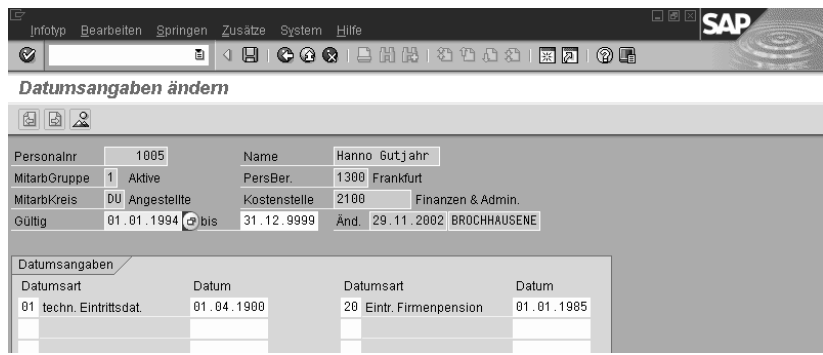


Abbildung 6.4 Datumsangaben in Infotyp 0041

Insgesamt sind in diesem Infotyp zwölf verschiedene Datumsarten möglich. Im Data Dictionary abgebildet sind diese Felder als eine Kette von Einzelfeldern, die sich durch die Nummern im Feldnamen unterscheiden. Abbildung 6.5 zeigt einen Ausschnitt aus der Datenstruktur des Infotyps 0041.

Datumsangaben

Der nachfolgend beschriebene Report zeigt die Verfahrensweise, wie auf diese Datenfelder zugegriffen werden kann. Es soll in diesem Beispiel die Zahl der Anspruchsjahre zum Monatsersten bestimmt werden. In der Liste sollen dabei das Eintrittsdatum und das Basisdatum (Datumsart 20) erscheinen. Die Anspruchsjahre werden als Differenz aus dem Selektionsdatum und dem jüngsten Datum aus Eintrittsdatum und Basisdatum berechnet. Es sollen nur die Personalnummern verarbeitet werden, bei

denen die Datumsart 20 zum Selektionszeitpunkt vorhanden ist. In Abbildung 6.6 ist die erzeugte Liste dargestellt.

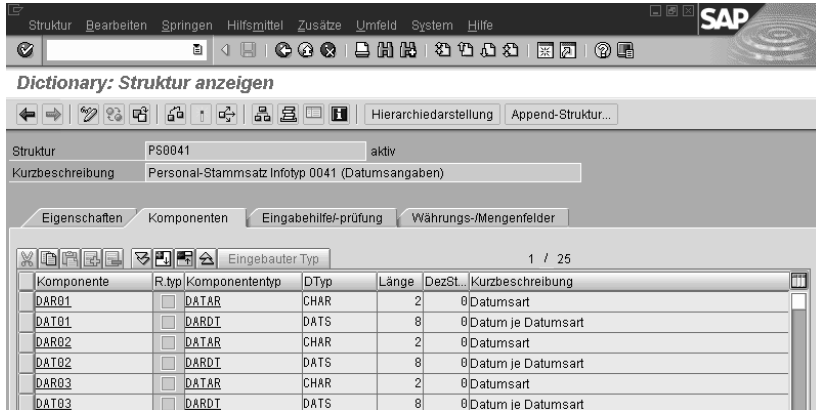


Abbildung 6.5 Struktur Infotyp 0041 (Datumsangaben)

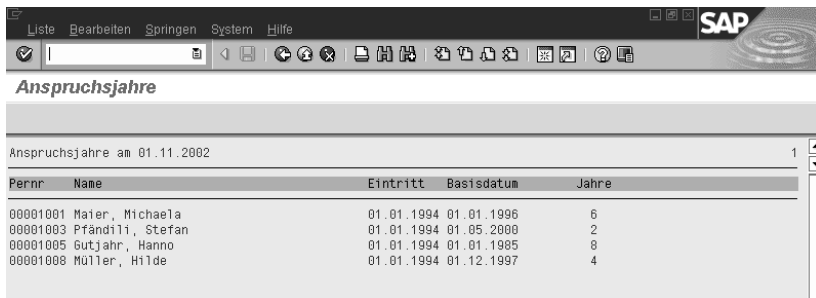


Abbildung 6.6 Listausgabe des Reports »Anspruchsjahre«

Daten-deklarationen

Am Anfang des Reports erfolgen zunächst die notwendigen Datendeklarationen. Für das Lesen der Wiederholungsstrukturen werden die Felder dar und dat benötigt.

```
REPORT yrhr_date LINE-SIZE 132.
TABLES: pernr.
INFOTYPES: 0002, 0041.
DATA: hire_date LIKE sy-datum, "Eintrittsdatum
      rel_date LIKE sy-datum, "Berechnungsbasis
      rel_years TYPE i, "Anspruchsjahre
      coname(40). "Nachname, Vorname
DATA: dar LIKE p0041-dar01, "Datumsart
      dat LIKE p0041-dat01. "Datum
```

Zum Zeitpunkt `INITIALIZATION` wird aus dem Tagesdatum der Monats-erste abgeleitet und für das Selektionsdynpro zur Verfügung gestellt. Die Systemvariable `syst-tvar0` wird für die Listüberschrift gefüllt.

```
INITIALIZATION.  
  pnpbegda = sy-datum.  
  pnpbegda+6(2) = '01'.  
START-OF-SELECTION.  
  WRITE pn-begda TO syst-tvar0 DD/MM/YYYY.
```

Zum Zeitpunkt `GET pernr` erfolgt wieder die Verarbeitung für jede Personalnummer. Zuerst werden alle Werte initialisiert, die im Report berechnet werden. Es wird geprüft, ob die interne Infotyptabelle `P0041` gefüllt ist. In älteren Releases muss hier statt Verwendung von `lines(p0041)` noch eine Hilfsvariable mit `DESCRIBE` versorgt werden. Die aktuell gültigen Werte des Infotyps `0041` werden mit dem Makro `RP_PROVIDE_FROM_LAST` in die Kopfzeile der internen Tabellen `P0041` gestellt. Ist kein gültiger Wert vorhanden, erhält das Feld `pnp-sw-found` einen Wert ungleich Null. Auch in diesem Fall erfolgt für die Personalnummer keine weitere Verarbeitung. Danach wird der aktuelle Wert des Infotyps `0002` gelesen. Der Name wird aus Nachname und Vorname zusammengesetzt. In den beiden Routinen `read_hire_date` und `read_p0041` werden die interessierenden Datumswerte gelesen. Je nach Ausprägung dieser Werte werden die Anspruchsjahre berechnet und in der Liste ausgegeben.

GET PERNR

```
GET pernr.  
  PERFORM clear_all.  
  CHECK lines( p0041 ) > 0.  
  rp_provide_from_last p0041 space pn-begda pn-begda.  
  CHECK pnp-sw-found <> 0.  
  rp_provide_from_last p0002 space pn-begda pn-begda.  
  PERFORM concat_name USING coname.  
  PERFORM read_hire_date USING hire_date.  
  PERFORM read_p0041 USING rel_date.  
  IF rel_date < hire_date.  
    rel_years = pn-begda+0(4) - hire_date+0(4).  
    IF hire_date+4(4) >= pn-begda+4(4).  
      rel_years = rel_years - 1.  
    ENDIF.  
  ELSE.
```

```

rel_years = pn-begda+0(4) - rel_date+0(4).
IF rel_date+4(4) => pn-begda+4(4).
    rel_years = rel_years - 1.
ENDIF.
ENDIF.
WRITE: / pernr-pernr,
        coname,
        hire_date,
        rel_date,
        rel_years.
END-OF-SELECTION.

```

In Routine `clear_all` werden die reportspezifischen Variablen initialisiert:

```

FORM clear_all.
    CLEAR: coname, rel_date, hire_date, rel_years.
ENDFORM.

```

Die Aufbereitung des Namens erfolgt in der Routine `concat_name`:

```

FORM concat_name USING p_conname.
    CONCATENATE p0002-nachn p0002-vorna
        INTO p_conname SEPARATED BY ', '.
ENDFORM.

```

Eintrittsdatum Das Eintrittsdatum wird mit dem Funktionsbaustein `HR_ENTRY_DATE` bestimmt, der bereits in Kapitel 3 vorgestellt wurde:

```

FORM read_hire_date USING p_hire_date.
    CALL FUNCTION 'HR_ENTRY_DATE'
        EXPORTING
            persnr                = pernr-pernr
        IMPORTING
            entrydate              = p_hire_date
        EXCEPTIONS
            entry_date_not_found = 1
            pernr_not_assigned   = 2
            OTHERS                = 3.
    IF sy-subrc <> 0.
*     ...

```



```
ENDIF.  
ENDFORM.
```

In der Routine `read_p0041` wird der Wert für die Datumsart 20 ermittelt. **DO-Schleife**
Mit der Anweisung

```
DO 12 TIMES  
  VARYING dar  
    FROM p0041-dar01 NEXT p0041-dar02
```

wird die Struktur `p0041` verarbeitet. Bei jedem Durchlaufen der **DO-Schleife** wird das Feld `dar` gefüllt. Der Aufsetzpunkt wird durch das Feld `p0041-dar01` bestimmt, die Schrittweite durch die Differenz zum Feld `p0041-dar02`. Analog wird beim Feld `dat` verfahren. Wenn keine Einträge mehr gefunden werden oder die Datumsart 20 gelesen worden ist, wird die Schleife verlassen:

```
FORM read_p0041 USING p_rel_date.  
  DO 12 TIMES  
    VARYING dar  
      FROM p0041-dar01 NEXT p0041-dar02  
    VARYING dat  
      FROM p0041-dat01 NEXT p0041-dat02  
    IF dar IS INITIAL.  
      EXIT.  
    ELSEIF dar = '20'.  
      p_rel_date = dat.  
      EXIT.  
    ENDIF.  
  ENDDO.  
ENDFORM.
```

Auswertung der Basisbezüge

Häufig ist die Ermittlung der Lohnarten und von deren Beträgen aus dem Infotyp 0008 (*Basisbezüge*) notwendig. Diese sind wie im vorangegangenen Beispiel in Wiederholungsgruppen des Infotyps 0008 gespeichert. Zudem werden die Beträge der indirekt bewerteten Lohnarten nicht in den Tabellen dauerhaft gespeichert, sondern erst zur Laufzeit ermittelt. Für die Auswertung dieser Lohnarten stehen Funktionsbausteine zur Verfügung, die die notwendige Funktionalität enthalten.

Entwicklung der Basisbezüge

In dem Beispielreport soll für die Mitarbeiter die Entwicklung der Basisbezüge der letzten zehn Jahre in einer Liste dargestellt werden. Als Selektionsbild wird das Standardselektionsbild verwendet. Als Vorschlagswerte werden das Jahresende und der Beginn des Auswertungszeitraums für den Datenauswahlzeitraum bestimmt (siehe Abbildung 6.7).

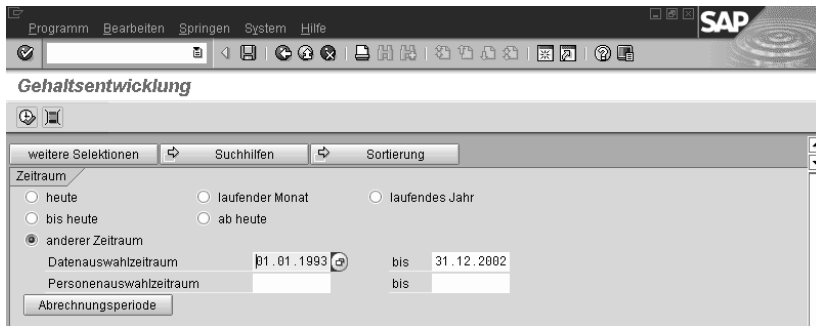


Abbildung 6.7 Selektionsbild des Reports »Gehaltsentwicklung«

In der Liste (siehe Abbildung 6.8) sollen neben Personalnummer und Name die in dem Zeitraum gültigen Gesamtbezüge dargestellt werden. Bei einem Wechsel der Währung sollen alle Beträge in die zuletzt gültige Währung umgerechnet werden. Zudem sollen die Zeilen mit den Namen farblich hervorgehoben werden.

Pernr	Name	Beginn	Ende	Betrag	Währ
00001003	Prändl11, Stefan	01.01.1994	31.03.1998	1.896,51	EUR
		01.04.1998	31.12.2001	1.955,77	EUR
		01.01.2002	31.12.2002	1.955,77	EUR
00001004	Paulsen, Olaf	01.01.1994	31.03.1998	1.656,79	EUR
		01.04.1998	30.04.1998	1.743,68	EUR
		01.05.1998	31.12.2001	1.834,78	EUR
		01.01.2002	30.09.2002	1.902,72	EUR
		01.10.2002	31.12.2002	2.068,98	EUR
00001005	Gutjahr, Hanno	01.01.1994	31.03.1998	2.132,09	EUR
		01.04.1998	31.12.2001	2.246,62	EUR
		01.01.2002	31.12.2002	2.246,62	EUR

Abbildung 6.8 Listausgabe des Reports »Gehaltsentwicklung«

Bewertete Lohnarten

Die Datendeklarationen umfassen neben den für den später verwendeten Funktionsbaustein RP_FILL_WAGE_TYPE_TABLE_EXT notwendigen Infotypen die internen Tabellen der bewerteten Lohnarten bewlart und t_

bewlart und den Arbeitsbereich w_bewlart. Außerdem wird eine Variable last_curr für die zuletzt verwendete Währung benötigt.

```
REPORT yrhr_p0008_list LINE-SIZE 132.
TABLES: pernr.
INFOTYPES: 0001, 0002, 0007, 0008.
DATA: bewlart TYPE TABLE OF pbwla, "Lohnarten lx
      t_bewlart TYPE TABLE OF pbwla, "Lohnarten gesamt
      w_bewlart TYPE pbwla, "Lohnarten Work
      last_curr LIKE p0008-waers, "Letzte Waehrung
      coname(40). "Nachname, Vorname
```

Zum Zeitpunkt INITIALIZATION werden die Vorschlagswerte für den Datenauswahlzeitraum pnpbegda und pnpendda ermittelt. **Datenauswahlzeitraum**

```
INITIALIZATION.
  pnpendda = sy-datum.
  pnpendda+4(4) = '1231'.
  PERFORM day_minus_years(sapfp500)
    USING pnpendda '10' pnpbegda.
  pnpbegda = pnpbegda + 1.
START-OF-SELECTION.
```

Am Anfang der Verarbeitung für jede Personalnummer wird die Initialisierung vorgenommen und der gültige Name aus dem Infotyp 0002 bestimmt. Die Lohnarten werden mit der Routine read_p0008 aus dem Infotyp 0008 gelesen. Die Ausgabe von Personalnummer und Name wird farblich durch die FORMAT-Anweisung aufbereitet. Danach werden die Lohnarten pro Infotypzeitraum in dem Unterprogramm summiert und ausgegeben.

```
GET pernr.
  PERFORM clear_all.
  rp_provide_from_last p0002 space pn-begda pn-endda.
  PERFORM concat_name USING coname.
  PERFORM read_p0008.
  FORMAT COLOR COL_GROUP INTENSIFIED ON.
  WRITE: / pernr-pernr,
         coname, 132 space.
  FORMAT COLOR OFF.
  PERFORM list_sum.
END-OF-SELECTION.
```

In der Routine `clear_all` werden die Variablen, Strukturen und internen Tabellen initialisiert:

```
FORM clear_all.  
  CLEAR: coname, bewlart, t_bewlart,  
        w_bewlart, last_curr.  
ENDFORM.
```

Der Name wird aus Nachname und Vorname gebildet:

```
FORM concat_name USING p_conname.  
  CONCATENATE p0002-nachn p0002-vorna  
    INTO p_conname SEPARATED BY ', '  
ENDFORM.
```

In dem Unterprogramm `read_p0008` wird in einer `PROVIDE`-Schleife die interne Infotypentabelle `p0008` für die Daten aus dem Datenauswahlzeitraum bearbeitet. Der Funktionsbaustein `RP_FILL_WAGE_TYPE_TABLE_EXT` füllt für den Betrachtungszeitraum die interne Tabelle der bewerteten Lohnarten `bewlart` aus den aktuellen Werten des Infotyps `0008`. In der internen Tabelle `t_bewlart` werden die Ergebnisse aller Perioden für eine Personalnummer gesammelt. Zudem wird die letzte Währung des Infotyps in der Variablen `last_curr` festgehalten.

```
FORM read_p0008.  
  PROVIDE * FROM p0008 BETWEEN pn-begda AND pn-endda.  
  CALL FUNCTION 'RP_FILL_WAGE_TYPE_TABLE_EXT'  
    EXPORTING  
      begda                = p0008-begda  
      endda                = p0008-endda  
      infity               = '0008'  
      pernr               = pernr-pernr  
  TABLES  
    pp0001                = p0001  
    pp0007                = p0007  
    pp0008                = p0008  
    ppbwla                = bewlart  
  EXCEPTIONS  
    error_at_indirect_evaluation = 1  
    OTHERS                      = 2.  
  IF sy-subrc <> 0.  
*   ...
```

```

ENDIF.
APPEND LINES OF bewlart TO t_bewlart.
ENDPROVIDE.
last_curr = p0008-waers.
ENDFORM.

```

In der Routine list_sum wird die interne Tabelle t_bewlart weiterverarbeitet. Wenn die Währung einer Lohnart in dieser Tabelle nicht mit der letzten Währung übereinstimmt, erfolgt die Umrechnung des Betrags mit dem Funktionsbaustein CONVERT_TO_LOCAL_CURRENCY und die Tabelle wird entsprechend geändert. Für gleiche Zeiträume werden die Werte summiert und in der Liste ausgegeben.

**Umrechnung der
Währung**

```

FORM list_sum.
  LOOP AT t_bewlart INTO w_bewlart.
    IF w_bewlart-waers <> last_curr.
      CALL FUNCTION 'CONVERT_TO_LOCAL_CURRENCY'
        EXPORTING
          date           = w_bewlart-endda
          foreign_amount = w_bewlart-betrg
          foreign_currency = w_bewlart-waers
          local_currency = last_curr
        IMPORTING
          local_amount   = w_bewlart-betrg
        EXCEPTIONS
          no_rate_found   = 1
          overflow        = 2
          no_factors_found = 3
          no_spread_found = 4
          derived_2_times = 5
          OTHERS          = 6.
      IF sy-subrc <> 0.
        ELSE.
          w_bewlart-waers = last_curr.
          MODIFY t_bewlart FROM w_bewlart.
        ENDIF.
      ENDIF.
    ENDIF.
  AT END OF endda.
  SUM.
  WRITE: /50 w_bewlart-begda, w_bewlart-endda,
         w_bewlart-betrg, last_curr.

```

ENDAT.
 ENDLOOP.
 ENDFORM.

6.1.2 Aufbau eines Reports für Infotypen der Zeitwirtschaft

Bei Verwendung der logischen Datenbank PNP werden zum Zeitpunkt GET PERNR für alle mit INFOTYPES deklarierten Infotypen nnnn die internen Tabellen Pnnnn gefüllt. Bei der großen Menge an Sätzen bei den Infotypen der Zeitwirtschaft ist dieses Vorgehen hier nicht zu empfehlen. Wie dann zu verfahren ist, wird an dem Report »Übersicht Abwesenheiten« für den Infotyp 2001 (*Abwesenheiten*) erläutert. Für alle ausgewählten Mitarbeiter sollen die in den Reportparametern angegebenen Abwesenheitsarten des laufenden Kalenderjahres ausgewertet werden. Dabei soll es möglich sein, mit einen Schalter die Auflistung der einzelnen Abwesenheitssätze an- und auszuschalten. Die Zusammenfassung der Abwesenheiten soll für die Personengruppen und -kreise getrennt erfolgen. Das Ergebnis ist in Abbildung 6.9 zu sehen.

The screenshot shows the SAP report 'Übersicht Abwesenheiten'. The main table lists absences for three employees: Zaucker, Ulrike; Paulsen, Olaf; and Gutjahr, Hanno. A summary table at the bottom shows the total days for each absence type: 1 DT 0100 Urlaub (14,00), 1 DU 0100 Urlaub (6,00), and 1 DU 0200 Krankheit mit Attest (9,00).

Pernr	Name	Beginn	Ende	Abwesenheitsart	Abwesenheitstage
00001002	Zaucker, Ulrike	01.07.2002	12.07.2002	0100 Urlaub	10,00
		07.10.2002	10.10.2002	0100 Urlaub	4,00
00001004	Paulsen, Olaf	02.10.2002	10.10.2002	0100 Urlaub	6,00
		01.03.2002	13.03.2002	0200 Krankheit mit Attest	9,00
00001005	Gutjahr, Hanno				
00001008	Müller, Hilde				
1 DT 0100 Urlaub					14,00
1 DU 0100 Urlaub					6,00
1 DU 0200 Krankheit mit Attest					9,00

Abbildung 6.9 Listausgabe des Reports »Abwesenheiten«

MODE n Bei der Datendeklaration ist beim Infotyp 2001 der Zusatz **MODE n** angegeben. Dieser Zusatz verhindert das Füllen der internen Tabelle P2001 zum Zeitpunkt GET PERNR. Für das Unterdrücken der Einzelliste ist der Parameter **s_name** eingerichtet, für die Auswahl der interessierenden Abwesenheiten die Selektion **s_abs**. Für die personenübergreifende Zusammenfassung wird die interne Tabelle **t_absence** mit dem Arbeitsbereich **w_absence** angelegt:

```

REPORT yrhr_p2001_list LINE-SIZE 132.
TABLES: pernr.
INFOTYPES: 0001, 0002,
            2001 MODE n.
PARAMETERS: s_name AS CHECKBOX.
SELECT-OPTIONS: s_abs FOR p2001-awart.
DATA: coname(40). "Name, Vorname
TYPES: BEGIN OF absence,
        persg LIKE p0001-persg, "Personengruppe
        persk LIKE p0001-persk, "Personenkreis
        awart LIKE p2001-awart, "Abwesenheitsart
        abwtg LIKE p2001-abwtg, "Abwesenheitsname
      END OF absence.
DATA: w_absence TYPE absence,
      t_absence TYPE TABLE OF absence.

```

Zum Zeitpunkt INITIALIZATION werden die Vorschlagswerte des Datenauswahlzeitraums ermittelt:

```

INITIALIZATION.
  pnpbegda = pnpendda = sy-datum.
  pnpbegda+4(4) = '0101'.
  pnpendda+4(4) = '1231'.
START-OF-SELECTION.

```

Für jede Personalnummer werden die bereits bekannten Verarbeitungen **GET PERNR** vorgenommen und der Infotyp 2001 wird mit der Routine `read_p2001` gelesen. Wenn der Parameter `s_name` aktiviert ist, wird die Einzelverarbeitung mit der Routine `list_name` durchgeführt. Nach der Verarbeitung aller Personalnummern wird die Zusammenfassung mit der Routine `list_sum` ausgegeben:

```

GET pernr.
  PERFORM clear_all.
  rp_provide_from_last p0002 space pn-begda pn-endda.
  PERFORM concat_name USING coname.
  PERFORM read_p2001.
  IF NOT s_name IS INITIAL.
    FORMAT COLOR COL_GROUP INTENSIFIED ON.
    WRITE: / pernr-pernr,
           coname, 132 space.
    FORMAT COLOR OFF.

```

```

        PERFORM list_name.
    ENDIF.
END-OF-SELECTION.
    PERFORM list_sum.

```

Die Verarbeitung der Routinen `clear_all` und `concat_name` erfolgt in der von den anderen Beispielen bereits bekannten Weise.

```

FORM clear_all.
    CLEAR: coname.
ENDFORM.
FORM concat_name USING p_conname.
    CONCATENATE p0002-nachn p0002-vorna
        INTO p_conname SEPARATED BY ', '.
ENDFORM.

```

Füllen der Tabelle p2001

Zu Beginn der Routine `read_p2001` werden nur diejenigen Zeilen der internen Tabelle `p2001` gefüllt, die durch den Datenauswahlzeitraum festgelegt worden sind. Dies erledigt der Funktionsbaustein `rp_read_all_time_ity`. Aus der internen Tabelle `p2001` werden dann die Einträge gelöscht, die aufgrund der Angaben in den Selektionsoptionen nicht verarbeitet werden sollen. Wenn die Tabelle jetzt noch Werte enthält, werden die Abwesenheiten weiterverarbeitet. Die korrespondierenden Inhalte des Infotyps `p0001` werden in den Arbeitsbereich `w_absence` übertragen. Das Gleiche geschieht mit den verbliebenen Einträgen der internen Tabelle `p2001`.

```

FORM read_p2001.
    rp_read_all_time_ity pn-begda pn-endda.
    DELETE p2001 WHERE NOT ( awart IN s_abs ).
    IF lines( p2001 ) > 0.
        MOVE-CORRESPONDING p0001 TO w_absence.
        LOOP AT p2001.
            MOVE-CORRESPONDING p2001 TO w_absence.
            COLLECT w_absence INTO t_absence.
        ENDLOOP.
    ENDIF.
ENDFORM.

```

Abwesenheits- arten

Für die Auflistung der einzelnen Abwesenheiten in dem Unterprogramm `list_name` wird innerhalb des Loops über die interne Tabelle `p2001` der Text der Abwesenheitsart mit der Routine `re554t` gelesen. Zur Vereinfachung wird angenommen, dass der zweite Parameter (Gruppierung der

Personalteilbereiche für Ab-/Anwesenheitsarten) den konstanten Wert »01« hat.

```
FORM list_name.  
  DATA: atext LIKE t554t-atext.  
  LOOP AT p2001.  
    PERFORM re554t  
      USING sy-langu '01' p2001-awart atext.  
    WRITE: /30 p2001-begda, p2001-enddda,  
      p2001-awart, atext, p2001-abwtg.  
  ENDLOOP.  
ENDFORM.
```

In der Routine re554t wird die Texttabelle T554T der Abwesenheitsarten **Texttabelle** in der Anmeldesprache gelesen und zur Verfügung gestellt:

```
FORM re554t USING p_langu p_moabw p_awart p_atext.  
  DATA: w_t554t LIKE t554t.  
  SELECT SINGLE * FROM t554t  
    INTO w_t554t  
    WHERE sprsl = p_langu  
      AND moabw = p_moabw  
      AND awart = p_awart.  
  IF sy-subrc = 0.  
    p_atext = w_t554t-atext.  
  ELSE.  
    CLEAR p_atext.  
  ENDIF.  
ENDFORM.
```

Die Zusammenfassung wird in dem LOOP über die Tabelle t_absence ausgegeben.

```
FORM list_sum.  
  DATA: atext LIKE t554t-atext.  
  SKIP 2.  
  LOOP AT t_absence INTO w_absence.  
    PERFORM re554t  
      USING sy-langu '01' w_absence-awart atext.  
    WRITE: / w_absence-persg, w_absence-persk,  
      w_absence-awart, atext, w_absence-abwtg.
```

```
ENDLOOP.  
ENDFORM.
```

6.2 Organisationsmanagement

Wie in Kapitel 3 dargestellt, sind bei der Verwendung der logischen Datenbank PCH sequenzielle und strukturelle Auswertungen möglich.

6.2.1 Sequenzielle Auswertungen

Stellen-
beschreibung

Die Anwendung einer sequenziellen Auswertung wird in dem folgenden Report näher erläutert. Die verschiedenen Objekttypen können im Infotyp 1002 eine verbale Beschreibung erhalten. In dem Beispiel soll die Beschreibung der Stelle (Objektyp C) zu einem Stichtag ausgegeben werden. Die Stellen-ID und die Stellenbezeichnung sollen farblich hervorgehoben werden (siehe Abbildung 6.10).

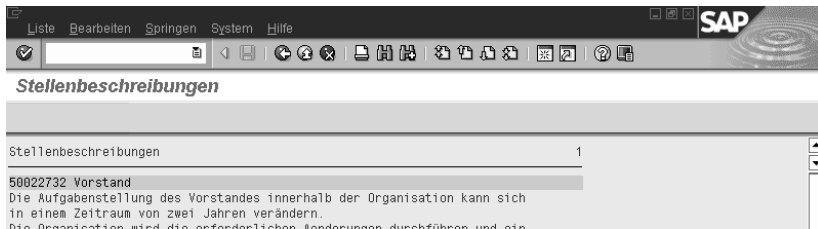


Abbildung 6.10 Sequenzielle Auswertung: Stellenbeschreibung

Im Deklarationsteil des Reports wird bei der TABLES-Anweisung *nicht* GDSTR angegeben. Hierdurch wird erkannt, dass *keine* strukturelle, sondern eine sequenzielle Auswertung erfolgen soll. Unter INFOTYPES wird der Infotyp 1002 (*Verbale Beschreibung*) deklariert. Dieser Infotyp ist als Tabelleninfotyp ausgeprägt. Deshalb ist zusätzlich die Definition der internen Tabelle `t_pt1002` erforderlich, die im weiteren Verlauf von einem Makro gefüllt wird.

```
REPORT ythr_org_seq LINE-SIZE 80.  
TABLES: objec.  
INFOTYPES: 1002.  
DATA: t_pt1002 TYPE TABLE OF pt1002 WITH HEADER LINE.
```

Stichtag Zum Zeitpunkt INITIALIZATION wird durch das Makro `rh_sel-key-date` erreicht, dass kein Auswahlzeitraum, sondern ein Stichtag in dem Selektionsbild angezeigt wird. Als Vorschlagswert für dieses Datum wird der 1. Januar des laufenden Jahres bereitgestellt. Weiterhin werden Vor-

schlagswerte für die Planvariante pchplvar und den Objekttyp pchotype erzeugt.

```
INITIALIZATION.  
  rh-sel-keydate.  
  pchplvar = '01'.  
  pchotype = 'C'.  
  pchobeg = sy-datum.  
  pchobeg+4(4) = '0101'.  
START-OF-SELECTION.
```

Bei GET objec stehen hier die Inhalte des Infotyps 1000 zum Stichtag zur Verfügung. Ausgegeben werden die Objekt-ID und die Bezeichnung des Objekts. **GET objec**

```
GET objec.  
  FORMAT COLOR COL_GROUP INTENSIFIED ON.  
  WRITE: / objec-objid, objec-stext, 80 space.  
  FORMAT COLOR OFF.
```

Bei der PROVIDE-Schleife über den Infotyp 1002 wird auf die Werte der Anmeldesprache eingeschränkt. Mit dem Makro rh-get-tbdat werden für den Tabelleninfotyp die Tabelleneinträge in die interne Tabelle t_pt1002 eingelesen. Der Inhalt dieser internen Tabelle kann dann in einer LOOP-Schleife ausgegeben werden. **Tabelleninfotyp**

```
PROVIDE * FROM p1002  
  BETWEEN pc-begda AND pc-endda  
  WHERE p1002-langu = sy-langu.  
  rh-get-tbdat p1002-infty p1002-tabnr t_pt1002.  
  LOOP AT t_pt1002.  
    WRITE: / t_pt1002-tline.  
  ENDLLOOP.  
ENDPROVIDE.
```

6.2.2 Strukturelle Auswertung

Bedeutend häufiger werden bei der Verwendung der logischen Datenbank PCH strukturelle Auswertungen durchgeführt.

Stellenbesetzungsplan

Es soll ein Stellenbesetzungsplan mit folgenden Informationen erstellt werden:

- ▶ Organisationseinheit, -bezeichnung
- ▶ Planstelle, Bezeichnung
- ▶ Personalnummer, Name (Nachname, Vorname)

Die Organisationseinheiten (Objekttyp »O«) werden ab Stelle 1 angedruckt. Entsprechend der Hierarchie werden Einrückungen vorgenommen. Die Planstellen (Objekttyp »S«) werden ab Stelle 20, die Personalnummern (Objekttyp »P«) ab Stelle 30 angedruckt. Für Organisationseinheiten, Planstellen und Personendaten werden unterschiedliche Hintergrundfarben verwendet.

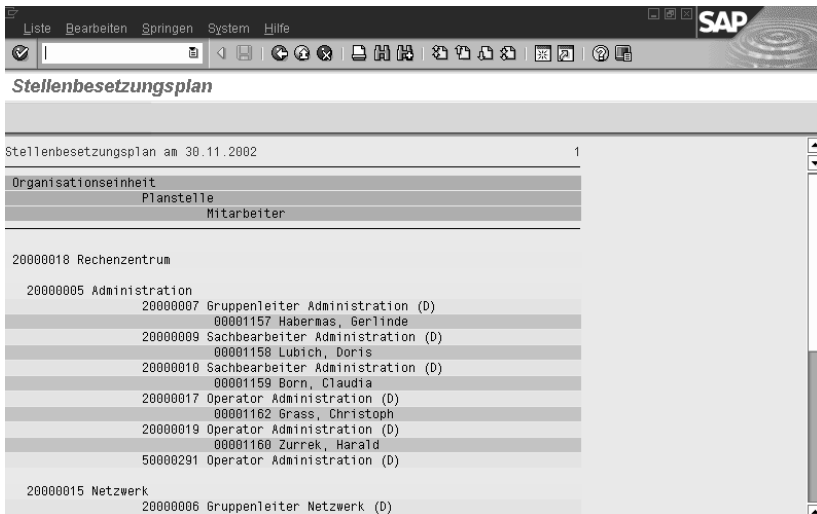


Abbildung 6.1 Strukturelle Auswertung: Stellenbesetzungsplan

Anzeige Auswertungsweg

Bei einer strukturellen Auswertung mit der logischen Datenbank PCH werden alle Objekte gelesen, die über den in den Selektionsparametern angegebenen Auswertungsweg erreichbar sind. Diese Auswertungswege werden im Customizing eingerichtet. In diesen Report soll der Auswertungsweg nicht bei den Selektionsparametern erscheinen, sondern fest vorgegeben werden und nicht änderbar sein. Aus diesem Grund ist es erforderlich, bei den Reporteigenschaften die Selektionsbildversion 900 (Strukturauswertung ohne Strukturparameter) zu nutzen.

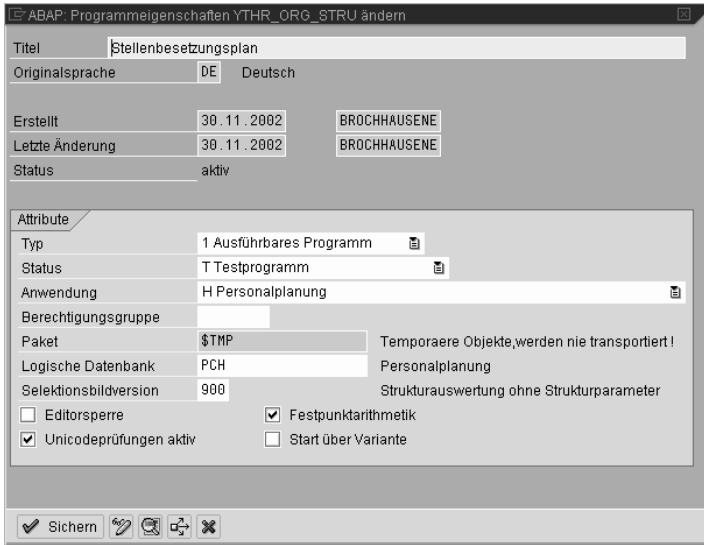


Abbildung 6.12 Eigenschaften des Reports »Stellenbesetzungsplan«

Im Deklarationsteil des Reports wird anhand der TABLES-Eintragung `gdstr` erkannt, dass eine strukturelle Auswertung vorliegt. Für den Ausdruck des Namens ist der Infotyp 0002 zu deklarieren. Für die Einrückung bei den Organisationseinheiten wird die Hilfsvariable `col` verwendet:

```
REPORT ythr_org_stru LINE-SIZE 80.
TABLES: objec,
        gdstr.
INFOTYPES: 0002.
DATA: col LIKE struc-level,
      coname(40).
```

Zum Zeitpunkt `INITIALIZATION` erfolgt neben den aus dem vorherigen Beispiel bekannten Anweisungen die Bestimmung des Auswertungswegs. Für die Überschrift wird die Systemvariable `sys-tvar0` mit dem Stichtagsdatum gefüllt:

```
rh-sel-keydate.
pchotype = '0'.
pchwegid = '0-S-P'.
START-OF-SELECTION.
WRITE pc-begda TO sys-tvar0 DD/MM/YYYY.
```

GET objec Bei GET *objec* werden jetzt alle Objekte auf dem Auswertungsweg mit ihren Infotypen zur Verfügung gestellt. Für die objektspezifische Aufbereitung wird auf den Objekttyp abgefragt. Bei dem Objekttyp »O« wird entsprechend der Auswertungsebene *struc-level* die Andruckposition bestimmt. Für die beiden Objekttypen »O« und »S« werden für die Ausgabe die Werte der Struktur *objec* verwendet.

```
GET objec.
  IF objec-otype = 'O'.
    col = struc-level * 2 .
    SKIP.
    FORMAT COLOR 2.
    WRITE: AT col objec-objid, objec-stext, 80 space.
  ENDIF.
  IF objec-otype = 'S'.
    FORMAT COLOR 3.
    WRITE: /20 objec-objid, objec-stext, 80 space.
  ENDIF.
```

Für den Personalstamm stehen die in *INFOTYPES* deklarierten internen Infotypentabellen zur Verfügung. Hier wird der Infotyp 0002 ausgewertet und der Name mit der Routine *concat_name* aufbereitet:

```
  IF objec-otype = 'P'.
    PROVIDE vorna nachn FROM p0002
      BETWEEN pc-begda AND pc-endda.
    PERFORM concat_name USING coname.
    FORMAT COLOR 4.
    WRITE: /30 objec-objid, coname, 80 space.
  ENDPROVIDE.
ENDIF.
END-OF-SELECTION.
FORM concat_name USING p_conname.
  CONCATENATE p0002-nachn p0002-vorna
    INTO p_conname SEPARATED BY ' ', '.
ENDFORM.
```

6.3 Abrechnungsdaten

In Abschnitt 3.5 wurde bereits der Zugriff auf die Abrechnungsdaten dargestellt. An dem folgenden Beispiel sollen weitere Funktionsbausteine, die die Auswertung der Daten der Personalabrechnung unterstützen, erläutert werden.

Als Ergebnis des Beispielreports sieht man die Liste in Abbildung 6.13. Aufgeführt sind unter den Namen der Mitarbeiter die Abrechnungsperioden und die gezahlten Nettogehälter.

The screenshot shows the SAP 'Übersicht Nettogehalt' report. It features a menu bar with 'Liste', 'Bearbeiten', 'Springen', 'System', and 'Hilfe'. Below the menu is a toolbar with various icons. The main content area is titled 'Übersicht Nettogehalt' and contains two data sections. The first section, labeled 'Übersicht Nettogehalt 1', shows a table with columns 'Pernr', 'Name', 'Periode', and 'Betrag'. The second section, labeled 'Übersicht Nettogehalt 2', shows a 'Statistik' table with rows for 'Selektierte Personen', 'Bearbeitete Personen', and 'Abgelehnte Personen'.

Pernr	Name	Periode	Betrag
00001001	Maier, Michaela	06 / 2002	1.992,66
		07 / 2002	1.553,25
		08 / 2002	1.553,25
00001007	Ulrich, Hanna	06 / 2002	1.774,99
		07 / 2002	1.441,55
		08 / 2002	1.441,55

Statistik	
Selektierte Personen	2
Bearbeitete Personen	2
Abgelehnte Personen	0

Abbildung 6.13 Report »Übersicht Nettogehalt«

Am Ende der Liste wird eine Zusammenfassung ausgegeben. Falls Fehler bei der Bearbeitung aufgetreten sind, wird eine Fehlerliste angezeigt (Abbildung 6.14).

The screenshot shows the SAP 'Übersicht Nettogehalt' report with error messages. It features a menu bar with 'Liste', 'Bearbeiten', 'Springen', 'System', and 'Hilfe'. Below the menu is a toolbar with various icons. The main content area is titled 'Übersicht Nettogehalt' and contains three data sections. The first section, labeled 'Übersicht Nettogehalt 2', shows a table with columns 'Pernr', 'Name', 'Periode', and 'Betrag'. The second section, labeled 'Übersicht Nettogehalt 3', shows a 'Fehlermeldungen' table with columns 'Pernr', 'Typ', and 'Fehlertext'. The third section, labeled 'Übersicht Nettogehalt 3', shows a 'Statistik' table with rows for 'Selektierte Personen', 'Bearbeitete Personen', and 'Abgelehnte Personen'.

Pernr	Name	Periode	Betrag
00001008			

Fehlermeldungen		
Pernr	Typ	Fehlertext
00001008	E	NO_RESULTS

Statistik	
Selektierte Personen	1
Bearbeitete Personen	0
Abgelehnte Personen	1

Abbildung 6.14 Fehlermeldungen im Report »Übersicht Nettogehalt«

**Auswertung der
Abrechnungsergebnisse**

Für die Auswertung der Abrechnungsergebnisse ist die Definition der internen Tabelle `t_result` mit dem Arbeitsbereich `w_result` erforderlich. Notwendig für die Bearbeitung der Lohnarten aus der Tabelle RT des Abrechnungsclusters ist die Deklaration des Arbeitsbereichs `w_rt`. Die weiteren Datendefinitionen werden für die Reportstatistik und die Ausgabe der Fehlertabelle benötigt.

```
REPORT ythr_pay_net LINE-SIZE 80.
TABLES: pernr.
INFOTYPES: 0002.
DATA: coname(40).
DATA: t_result TYPE TABLE OF payde_result,
      w_result TYPE payde_result,
      w_rt TYPE pc207.
DATA: g_select TYPE i,
      g_proces TYPE i,
      g_reject TYPE i.
DATA: error_int TYPE TABLE OF hrerror.
```

Am Anfang der Verarbeitung müssen das ABAP Memory für die Fehlerliste und die Reportstatistik mit den beiden aufgeführten Funktionsbausteinen initialisiert werden:

```
START-OF-SELECTION.
  CALL FUNCTION 'HR_REFRESH_ERROR_LIST'.
  CALL FUNCTION 'HR_REFRESH_STAT_LIST'.
```

GET pernr Zum Zeitpunkt `GET pernr` werden zunächst bereits bekannte Verarbeitungen durchgeführt. Die Anzahl der selektierten Mitarbeiter wird in der Variablen `g_select` fortgeschrieben:

```
GET pernr.
  rp_provide_from_last p0002 space pn-begda pn-begda.
  PERFORM concat_name USING coname.
  FORMAT COLOR COL_GROUP INTENSIFIED ON.
  WRITE: / pernr-pernr,
         coname, 80 space.
  FORMAT COLOR OFF.
  g_select = g_select + 1.
```


Der Import der aktuellen Abrechnungsergebnisse des Datenauswahlzeitraums in die Tabelle `t_result` wird mit dem Funktionsbaustein `HR_GET_PAYROLL_RESULTS` durchgeführt:

```
CALL FUNCTION 'HR_GET_PAYROLL_RESULTS'
  EXPORTING
    pernr                = pernr-pernr
    pabrj                = pn-begda(4)
    pabrp                = pn-begda+4(2)
    pabrj_end            = pn-endda(4)
    pabrp_end            = pn-endda+4(2)
    actual                = 'A'
  TABLES
    result_tab           = t_result
  EXCEPTIONS
    no_results           = 1
    error_in_currency_conversion = 2
    t5001_entry_not_found = 3
    period_mismatch_error = 4
    t549q_entry_not_found = 5
    internal_error       = 6
    wrong_structure_of_result_tab = 7.
```

Im Fehlerfall wird der Funktionsbaustein `HR_APPEND_ERROR_LIST` aufgerufen, der die Fehlerliste `error_int` im ABAP Memory mit der ID »HRERRLIST« fortschreibt. Für jede Personalnummer erfolgt die Ausgabe der Liste der Entwicklung des Nettoentgelts mit der Routine `list`. **Fehlerliste**

```
IF sy-subrc NE 0.
  CALL FUNCTION 'HR_APPEND_ERROR_LIST'
    EXPORTING
      pernr = pernr-pernr
      arbgb = sy-msgid
      msgty = 'E'
      msgno = sy-msgno
      msgv1 = sy-msgv1
      msgv2 = sy-msgv2
      msgv3 = sy-msgv3
      msgv4 = sy-msgv4.
  g_reject = g_reject + 1.
ELSE.
```

```

    g_proces = g_proces + 1.
ENDIF.
PERFORM list.
END-OF-SELECTION.

```

Nach Ende der Selektion wird die Fehlertabelle `error_int` aus dem ABAP Memory gelesen und mit dem Funktionsbaustein `HR_DISPLAY_ERROR_LIST` ausgegeben.

```

NEW-PAGE.
IMPORT error_int FROM MEMORY ID 'HRERRLIST'.
CALL FUNCTION 'HR_DISPLAY_ERROR_LIST'
  EXPORTING
    no_popup      = 'X'
    no_print      = ' '
  TABLES
    error         = error_int
  EXCEPTIONS
    invalid_linesize = 1
    OTHERS         = 2.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
    WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ENDIF.

```

Reportstatistik Die Verarbeitung und Ausgabe der Reportstatistik wird von den Funktionsbausteinen `HR_APPEND_STAT_LIST` und `HR_DISPLAY_STAT_LIST` durchgeführt:

```

CALL FUNCTION 'HR_APPEND_STAT_LIST'
  EXPORTING
    selected = g_select
    processed = g_proces
    rejected = g_reject.
CALL FUNCTION 'HR_DISPLAY_STAT_LIST'
  EXPORTING
    no_popup      = 'X'
    no_print      = ' '
  EXCEPTIONS
    invalid_linesize = 1
    OTHERS         = 2.
IF sy-subrc <> 0.

```

```

MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
        WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ENDIF.

```

Die Aufbereitung des Namens erfolgt mit der Routine `concat_name`:

```

FORM concat_name USING p_conname.
    CONCATENATE p0002-nachn p0002-vorna
        INTO p_conname SEPARATED BY ', '.
ENDFORM.

```

Die Ausgabe der Abrechnungsergebnisse wird pro Abrechnungsperiode in dem LOOP über die interne Tabelle `t_result` durchgeführt. Die Werte der Lohnartentabelle RT werden einer tieferen Ebene in den Arbeitsbereich `w_rt` eingelesen. Das Nettoentgelt ist in der technischen Lohnart »/550« zu finden.

Lohnartentabelle

```

FORM list.
    LOOP AT t_result INTO w_result.
        WRITE : /40 w_result- evp-fpper+4(2) ,
            '/', w_result- evp-fpper(4).
        LOOP AT w_result-inter-rt INTO w_rt
            WHERE lgart = '/550'.
            WRITE: w_rt-betrg.
        ENDLLOOP.
    ENDLLOOP.
ENDFORM.

```

Bei der Auswertung der Ergebnisse wird hier vereinfachend davon ausgegangen, dass dieselben Lohnarten nicht mit mehreren Zeitraumkennzeichen vorliegen. Dieser Sachverhalt ist noch entsprechend zu berücksichtigen.

6.4 Zeitwirtschaft

In der Zeitwirtschaft sind die Ergebnisse der Zeitauswertung wie die Abrechnungsergebnisse in Clustern gespeichert. Wie diese zu lesen sind, wird in diesem Kapitel dargestellt. In der Abbildung 6.15 sieht man die Liste der erzeugten Zeitpaare und der Zeitart »0003«, Rahmenzeit. Das Ende jeder Woche wird durch eine horizontale Linie verdeutlicht.

Ergebnisse der Zeitauswertung

Pernr	Name	Datum	von	bis	Rahmenzeit
00001047	Fischer, Klaus				
		03.06.2002	07:51:08	17:04:11	8,07
		04.06.2002	07:51:26	17:14:53	8,25
		05.06.2002	07:59:44	17:04:29	8,07
		06.06.2002	07:42:44	17:08:06	8,14
		07.06.2002	07:41:51	13:19:52	5,33
		10.06.2002	07:41:50	17:18:09	8,30
		11.06.2002	07:44:16	17:00:42	8,01
		12.06.2002	07:48:16	17:00:09	8,00
		13.06.2002	07:59:32	17:03:59	8,07
		14.06.2002	07:41:05	13:06:13	5,10
		17.06.2002	07:44:28	17:04:30	8,08

Abbildung 6.15 Report der Zeitauswertung

Datendeklaration Bei der Datendeklaration sind die notwendigen Includes für die Pufferung und die Datenbeschreibungen der Cluster aufgeführt. Die Konventionen hierfür wurden in Abschnitt 3.5 dargestellt.

```
REPORT yrhr_time LINE-SIZE 80.
TABLES: pernr,
        pcl1,
        pcl2.
INFOTYPES: 0002.
INCLUDE rpppxd00.           "Daten fuer PCL1/2-Puffer
DATA: BEGIN OF COMMON PART buffer.
INCLUDE rpppxd10.
DATA: END OF COMMON PART.
INCLUDE: rpc2b200.
DATA: coname(40),          "Nachname, Vorname
      odate LIKE pt-ldate,  "Wochenerster
      odate_old LIKE pt-ldate. "Wochenerster alt
```

Vorschlagswerte Als Vorschlagswerte für den Datenauswahlzeitraum werden Monatserster und Monatsletzter des Vormonats erzeugt. Für die Listüberschrift werden die Systemvariablen `sys-tvar0` und `sys-tvar1` gefüllt.

```
INITIALIZATION.
  pnpbegda = sy-datum.
  pnpbegda+6(2) = '01'.
  PERFORM day_minus_months(sapfp500)
    USING pnpbegda '1' pnpbegda.
```

```

PERFORM last_day_in_month(sapfp500)
  USING pnpbegda pnpendda.
START-OF-SELECTION.
  WRITE pn-begda TO syst-tvar0 DD/MM/YYYY.
  WRITE pn-endda TO syst-tvar1 DD/MM/YYYY.

```

Neu zum Zeitpunkt GET pernr im Vergleich zu den bisher behandelten Beispielen ist die Routine read_time, die die Daten der Zeitauswertung listet und ausgibt:

```

GET pernr.
  PERFORM clear_all.
  rp_provide_from_last p0002 space pn-begda pn-begda.
  PERFORM concat_name USING coname.
  FORMAT COLOR COL_GROUP INTENSIFIED ON.
  WRITE: / pernr-pernr,
         coname, 80 space.
  FORMAT COLOR OFF.
  PERFORM read_time.
END-OF-SELECTION.

```

Das Include rpppxm00 enthält die Standardroutinen für die Pufferung der Daten bei der Benutzung von Makros:

```

INCLUDE rpppxm00.

```

Die Routine concat_name bereitet den Namen auf:

```

FORM concat_name USING p_conname.
  CONCATENATE p0002-nachn p0002-vorna INTO p_conname
SEPARATED BY ', '.
ENDFORM.

```

In der Routine read_time wird der Key für das Lesen des Clusters B2 der Clustertabelle PCL2 gefüllt. Das Lesen des Clusters erfolgt mit dem Makro rp-imp-c2-b2. Falls der Import erfolgreich war, werden die Zeitpaare aus der importierten Tabelle PT gelesen und ausgegeben. Bei einem Wochenwechsel, der mit der Routine get_first_day_in_week erkannt wird, erfolgt die Ausgabe einer horizontalen Linie. Für jeden Tag wird die Zeitart »0003« (Rahmenzeit) aus der importierten Tabelle ZES angelistet. **Cluster B2**

```

FORM read_time.
  b2-key-pernr = pernr-pernr .      "Personalnummer

```

```

b2-key-pabrp = pnpbegda+4(2).    "Abrechnungsperiode
b2-key-pabrj = pnpbegda(4).      "Abrechnungsjahr
b2-key-cltyp = '1'.              "original
rp-imp-c2-b2.
IF sy-subrc EQ 0.
  LOOP AT pt.
    PERFORM get_first_day_in_week(sapfp500)
      USING pt-ldate '01' odate.
    IF ( NOT odate_old IS INITIAL )
      AND ( odate_old <> odate ).
      NEW-LINE.
      ULINE AT 35.
    ENDIF.
    odate_old = odate.
    WRITE: /35(12) pt-ldate,
      (10) pt-begtm USING EDIT MASK '__:__:__',
      (10) pt-endtm USING EDIT MASK '__:__:__'.
    LOOP AT zes WHERE reday = pt-ldate+6(2)
      AND zstart = '0003'.
      WRITE: zes-anzhl.
      EXIT.
    ENDLLOOP.
  ENDLLOOP.
ELSE.
* ...
ENDIF.
ENDFORM.

```

6.5 Aufbereitung der Ausgabe mit dem ABAP List Viewer

Die Aufbereitung der in den bisher in diesem Kapitel dargestellten Listen kann durch den Einsatz des ABAP List Viewers (ALV) um weitere Funktionen ergänzt werden. Zum Beispiel können Sortierungen der angezeigten Liste oder Downloads nach Excel durchgeführt werden. Die Anzeige kann individuell konfiguriert und als Anzeigevariante abgespeichert werden. In Abbildung 6.16 wird die aus Abschnitt 6.1 bekannte Liste mit dem Funktionsbaustein REUSE_ALV_LIST_DISPLAY aufbereitet.

PersNr	Name	Bankkonto	Bankschl.	Name des Geldinstituts
1001	Maier, Michaela	1258693	20050000	Deutsche Bank Hamburg
1002	Zaucker, Ulrike	253571587	23984899	Vereinsbank München
1003	Pfändlil, Stefan	258369741	12388823	Deutsche Bank Paderborn
1004	Paulsen, Olaf	741852963	20050000	Deutsche Bank Hamburg
1005	Gutjahr, Hanno	258963147	23984899	Vereinsbank München
1006	Awad, Yasmin			
1007	Ulrich, Hanna	582369741	23022200	Volksbank Marschfelden
1008	Müller, Hilde	123456987	32902212	Raiffeisenbank Breitenrunn
1009	Braunstein, Herbert	564789123	10050033	Dresdner Bank Berlin
1010	Schmidtrohr, Frank			
1011	Förster, Claudia	123123	67292200	Volksbank Wiesloch
1014	Hintze, Gudrun	12581541	20050000	Deutsche Bank Hamburg
1015	Rickes, Alexander	1569874256	23022200	Volksbank Marschfelden
1016	Kaufman, Mike	21343214	12388823	Deutsche Bank Paderborn
1017	Sturm, Annette	1234145252	23022200	Volksbank Marschfelden

Abbildung 6.16 Ausgabe der Liste mit REUSE_ALV_LIST_DISPLAY

Eine weitere Möglichkeit bietet der Funktionsbaustein REUSE_ALV_GRID_DISPLAY. Das Ergebnis ist in Abbildung 6.17 zu sehen.

Bankverbindungen der Mitarbeiter am 31.12.2002				
PersNr	Name	Bankkonto	Bankschl.	Name des Geldinstituts
1001	Maier, Michaela	1258693	20050000	Deutsche Bank Hamburg
1002	Zaucker, Ulrike	253571587	23984899	Vereinsbank München
1003	Pfändlil, Stefan	258369741	12388823	Deutsche Bank Paderborn
1004	Paulsen, Olaf	741852963	20050000	Deutsche Bank Hamburg
1005	Gutjahr, Hanno	258963147	23984899	Vereinsbank München
1006	Awad, Yasmin			
1007	Ulrich, Hanna	582369741	23022200	Volksbank Marschfelden
1008	Müller, Hilde	123456987	32902212	Raiffeisenbank Breitenrunn
1009	Braunstein, Herbert	564789123	10050033	Dresdner Bank Berlin
1010	Schmidtrohr, Frank			
1011	Förster, Claudia	123123	67292200	Volksbank Wiesloch
1014	Hintze, Gudrun	12581541	20050000	Deutsche Bank Hamburg
1015	Rickes, Alexander	1569874256	23022200	Volksbank Marschfelden
1016	Kaufman, Mike	21343214	12388823	Deutsche Bank Paderborn

Abbildung 6.17 Ausgabe der Liste mit REUSE_ALV_GRID_DISPLAY

Für die Verwendung des ALV ist die Deklaration des TYPE-POOLS: `slis` **TYPE-POOLS** notwendig. Die im Report definierte Struktur der Ausgabe `display` wird verwendet, um die interne Tabelle `t_display` und den Arbeitsbereich `w_display` zu definieren. Weiterhin werden der Feldkatalog `alv_fieldcat` und Angaben für das Layout und die Variantenspeicherung definiert:

```
REPORT yrhr_bank_alv.
TYPE-POOLS: slis.
```

```

TABLES: pernr.
INFOTYPES: 0002, 0009.
DATA: coname(40),      "Name, Vorname
      bankname(40).   "Bankname
TYPES: BEGIN OF display,
      pernr LIKE pernr-pernr,
      coname(40),
      bankn LIKE p0009-bankn,
      bankl LIKE p0009-bankl,
      bankname(40),
      END OF display.
DATA: t_display TYPE TABLE OF display,
      w_display TYPE display.
DATA: alv_fieldcat TYPE slis_t_fieldcat_alv,
      w_alv_fieldcat TYPE LINE OF slis_t_fieldcat_alv,
      alv_layout TYPE slis_layout_alv,
      alv_variant TYPE disvariant.
INITIALIZATION.
      PERFORM last_day_in_month(sapfp500)
      USING sy-datum pnpbegda.
START-OF-SELECTION.

```

GET pernr Zum Zeitpunkt GET pernr werden die bekannten Verarbeitungen durchgeführt. Im Unterschied zu der einfachen Liste wird die Ausgabe in die interne Tabelle t_display unter Verwendung des Arbeitsbereiches w_display vorgenommen. Zur Verdeutlichung wird hier die Zuweisung für jedes einzelne Feld statt MOVE-CORRESPONDING verwendet:

```

GET pernr.
      PERFORM clear_all.
      rp_provide_from_last p0002 space pn-begda pn-begda.
      rp_provide_from_last p0009 '0' pn-begda pn-begda.
      PERFORM concat_name USING coname.
      PERFORM read_bankname USING bankname.
      MOVE: pernr-pernr TO w_display-pernr,
            coname TO w_display-coname,
            p0009-bankn TO w_display-bankn,
            p0009-bankl TO w_display-bankl,
            bankname TO w_display-bankname.
      APPEND w_display TO t_display.
END-OF-SELECTION.

```


Am Ende der Verarbeitung ist die Tabelle `t_display` gefüllt und kann mit **Ausgabe** der Routine `show_liste` ausgegeben werden:

```
PERFORM show_liste.
```

Die unten aufgeführten Routinen sind bereits bekannt.

```
FORM clear_all.  
  CLEAR: coname, bankname.  
ENDFORM.  
FORM concat_name USING p_conname.  
  CONCATENATE p0002-nachn p0002-vorna  
    INTO p_conname SEPARATED BY ', '  
ENDFORM.  
FORM read_bankname USING p_bankname.  
  DATA : bankdata TYPE bnka_bf.  
  CALL FUNCTION 'HRCA_READ_BANK_ADDRESS_2'  
    EXPORTING  
      bank_country = p0009-banks  
      bank_number  = p0009-bankl  
    IMPORTING  
      bank_data    = bankdata  
    EXCEPTIONS  
      not_found    = 1  
      OTHERS       = 2.  
  IF sy-subrc = 0.  
    p_bankname = bankdata-banka.  
  ELSE.  
    CLEAR p_bankname.  
  ENDIF.  
ENDFORM.
```

Neu ist die Verarbeitung in der Routine `show_liste`. Für die Anzeige des Titels werden die Variablen `grid_titel` und `list_begda` definiert und gefüllt:

```
FORM show_liste.  
  DATA: grid_title TYPE lvc_title,  
        list_begda(10).  
  WRITE pn-begda TO list_begda DD/MM/YYYY.  
  CONCATENATE 'Bankverbindungen der Mitarbeiter am'  
    list_begda INTO grid_title SEPARATED BY space.
```

Layout Für das gewünschte Layout werden die geeigneten Übergabeparamter der Struktur `alv_layout` angegeben.

```
alv_layout-colwidth_optimize = 'X'.  
alv_layout-zebra = 'X'.
```

Feldkatalog Für die einzelnen Spalten der Ausgabe werden die speziellen Eigenschaften definiert und in den Feldkatalog aufgenommen. Die Bedeutung der Felder ist der Dokumentation des weiter unten aufgerufenen Funktionsbausteins zu entnehmen.

```
CLEAR w_alv_fieldcat.  
w_alv_fieldcat-fieldname = 'PERNR'.  
w_alv_fieldcat-ref_tabname = 'PERNR'.  
w_alv_fieldcat-key = 'X'.  
APPEND w_alv_fieldcat TO alv_fieldcat.  
CLEAR w_alv_fieldcat.  
w_alv_fieldcat-fieldname = 'CONAME'.  
w_alv_fieldcat-reptext_ddic = 'Name'.  
APPEND w_alv_fieldcat TO alv_fieldcat.  
CLEAR w_alv_fieldcat.  
w_alv_fieldcat-fieldname = 'BANKN'.  
w_alv_fieldcat-ref_tabname = 'P0009'.  
APPEND w_alv_fieldcat TO alv_fieldcat.  
CLEAR w_alv_fieldcat.  
w_alv_fieldcat-fieldname = 'BANKL'.  
w_alv_fieldcat-ref_tabname = 'P0009'.  
APPEND w_alv_fieldcat TO alv_fieldcat.  
CLEAR w_alv_fieldcat.  
w_alv_fieldcat-fieldname = 'BANKNAME'.  
w_alv_fieldcat-ref_tabname = 'BNKA_BF'.  
w_alv_fieldcat-ref_fieldname = 'BANKA'.  
APPEND w_alv_fieldcat TO alv_fieldcat.
```

Nachdem der Feldkatalog vollständig gefüllt worden ist, kann jetzt der Funktionsbaustein `REUSE_ALV_GRID_DISPLAY` aufgerufen werden und die Ausgabe übernehmen. In der Parameterliste sind auch die Parameter `i_save` und `is_variant` aufgeführt, die für die Abspeicherung der Listvarianten erforderlich sind.

```
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'  
EXPORTING
```

```

i_callback_program = 'YTHR_BANK_ALV'
i_grid_title       = grid_title
is_layout         = alv_layout
it_fieldcat       = alv_fieldcat
i_save            = 'A'
is_variant        = alv_variant
TABLES
  t_outtab         = t_display
EXCEPTIONS
  program_error   = 1
  OTHERS          = 2.
IF sy-subrc <> 0.
  WRITE: / 'Fehler bei Aufbereitung'.
ENDIF.
ENDFORM.          "show_liste

```

Ähnliche Aufbereitungsmöglichkeiten bietet der Funktionsbaustein HR_ALV_LIST_DISPLAY. Dieser Funktionsbaustein kapselt den Aufruf des Funktionsbausteins REUSE_ALV_LIST_DISPLAY für Anwendungen im HR.

Index

A

ABAP List Viewer 280, 298, 340
Ablauf der Berechtigungsprüfung 175
Abrechnungsdaten 272
Abrechnungsergebnisse 21, 92, 125, 274
 Auswertung 94
 Lesen 125
Abrechnungsinfortypen 333
 anlegen 335
Abrechnungslohnarten, definieren 334
Abrechnungsprogramm 125
Abrechnungstatus 112
Abwesenheitsart 266
ADD_FIELDS_SPLIT_DEP 331
Ad-hoc-Query 287, 288, 339
 aufrufen 340
 erstellen 342
 Mengenoperationen 344
 Selektion 343
Adressänderung 155
AGate 392
Aktualitätskennzeichen 128
Anwendungskomponente 129
Anzeigtiefe 164
Argument, variables 249
asymmetrisch 154
Aufbauorganisation 161
Aufbauvorschrift 159
Aufgabenbeschreibungen 139
Auswertung
 sequenziell 101, 102, 268
 strukturell 101, 103, 269, 270
Auswertungsweg 98, 100, 101, 162, 270
Auswertungszeitraum 100
authority-check 138

B

BAdI 131
 Aufruf 133
 Coding 202
 Filter 134
 Implementierung 134, 202
BAdI-Builder 201

BAdI-Klasse 132
BAPI 367
 Aufruf 368, 372
 Explorer 369
 Verwendung 371
Basisbezüge 259
Belegauswertung 152
Benutzergruppen 296, 308
Benutzermenü, Berichte 289
Berechtigungsfelder 137
Berechtigungsgenerierung 141
Berechtigungshauptschalter 168
Berechtigungskonzept 137
Berechtigungslevel 153
Berechtigungsobjekt 137
Berechtigungsprofile 139
Berechtigungsprüfung 81, 104, 105
Bereichsmenüpflege 289
Bewegungsdaten 21
Bildsteuerung 214
BL_ALLOW_DUP_LINES 325
Business Add-In 129, 131, 182
Business Application Programming Interface → BAPI
Business Information Warehouse 288
Business Object Repository (BOR) 369
Business-Objekttyp 367

C

Cascading Style Sheets 418
CASE_SENSITIVE_SEL 330
CE-Modus 96, 97
Cluster 123
 B1 67
 B2 279
 Berechtigung 124
 EXPORT 123
 IMPORT 123
 Makro 123
 Pufferung 124
 TX 123
Clusterdirectory 77, 125
Clusterstruktur 71
Clustertabelle 92

Concurrent Employment 96
Content Provider 437
Customer-Exit 129
Customer-Function 197
Customizing, L&W-Events 450

D

DATA_REQUIRED 329
Datenauswahlzeitraum 84, 88, 261
Datenbankprogramm 82, 83
Datenbanktabelle PAnnnn 42
Datenbanktabelle PCL2 68, 70
Datenbeschaffung 88
Datenmodell des Organisationsmanagements 46
Datumsangaben 255
DBPNPCOM 107
DBPNPMAC 107
Definitionsschicht 131
Designanpassungen 418
dezentrale Zeiterfassung 157
Download 280
Dual-Host-Installation 393
Dynpro 34

E

Easy Web Transaction 405
Einführungsleitfaden 23, 31, 49, 51
Eintrittsdatum 116, 258
Employee Self-Service 387
Entscheidungsbaum 32
Ergebnisdaten 21
Erweiterung, zuordnen 205
Erweiterungsprojekt 198
ESS 387
 Kiosksysteme 423
 MOLGA 425
ESS-Szenario 155
Excel 280
externe Objekttypen 48, 59

F

Fehlerliste 275
Fehlertabelle 274
Feldattribut 35
Felder, kundeneigene 203
Feldkatalog 284

FlowLogic 406, 410
 Beispiel 411
 Events 414
Form Painter 362
Formular
 Drucken 364
Formulare 347
 Attribute 360
 erstellen 348
 Metadaten anlegen 349
Formularschnittstelle 361
freie Abgrenzung 86, 87
Funktion 241, 244
 kundenspezifisch 245
Funktionsbaustein 114
 Funktion Builder 114
 HR 115
 Suche 116
Funktionsexit 195
Für-Periode 95

G

GDSTR 102, 268
GET GROUP 97
GET OBJEC 102, 103
GET PAYROLL 92, 95
GET PERAS 96
GET PERNR 81, 88, 95
GET PERNR LATE 96
GET PERSON 97
globale Organisationsebene 141
globaler Bereich 296

H

HIGH-DATE 107
HIS 287, 291
 Aufruf 291
 Customizing 294
HR Business Content 288
HR_ALV_LIST_DISPLAY 285
HR_ENTRY_DATE 258
HR_FEATURE_BACKFIELD 118
HR_FEATURE_BACKTABLE 118
HR_GET_PAYROLL_RESULTS 275
HR-Formular-Workplace 347, 357
HR-Metadaten-Workplace 349, 350
HR-Report 150

HRUSER 418
HR-Zusatzfelder 317
HTML-Business-Funktionen 403
HTML-Templates 407
Human Resources Information
System, HIS 287

I

IAC 406
IGNORE_WAGE_TYPE_OPERA 330
IMG 129
Implementierungsschicht 131
Import 279
InfoSet Query 287, 339
 aufrufen 340
InfoSets 296
 anlegen 305
 Anlegen von Zusatzfeldern 313
 übergreifende Zusatzfelder 317
Infotyp 22, 24, 46
 ändern 111
 anlegen 230
 Bildmodifikation 32
 Datenstruktur 36
 Erweiterung 195, 226
 externe Infotypen 60
 Header 30
 Infotyp-View 44
 Konsistenzcheck 63
 Listbild 29
 Merkmal 32
 Objektidentifikation 26
 Organisationsmanagement 225
 SPA/GPA-Parameter 30
 Steuerungstabellen 51
 Struktur des Merkmals 33
 Subtyp 25, 41
 Tabelleninfotypen 57
 Vorschlagwert 29
 Zeitwirtschaft 238, 264
Infotypen-Menüs 45
INFOTYPES 88
 AS PERSON TABLE 97
 MODE N 88
Interface FI 254
Interface Toolbox 367, 375
 File-Layout 381

Interface-Format 379
Interface-Format 379
interne Objekttypen 47
Internet Transaction Server 392
Interpretation der zugeordneten
 Personalnummer 155
ITS 392
 Dateien 394
 Instanzen 393
 Programmiermodelle 405
 Serviceparameter 398
iViews 388

J

Join 91
 Teilintervall 92

K

Kiosksysteme 423
Knowledge Provider 437
Konsistenz des Datenmodells 61
Kontext 165
kontextabhängige strukturelle Berech-
 tigungsprüfung 168
Kontraktion 90
kundeneigene Einstellungen 63
kundeneigene Version von PA30 181
kundeneigenes Berechtigungsobjekt
 147, 177
Kundenobjekte 129
Kundenreportklassen 86
kundenspezifische Erweiterungen 41

L

L&W-Events 388, 435, 436
 Customizing 450
 Erstellung von 456
LAST_RECORD_ONLY 326
Layout 284
Lebens- und Arbeitsereignisse 435
Listüberschrift 253
Logische Datenbank 81
 PCH 98
 PNP 83
 PNPCE 96
Logische Struktur Pnnnn 57
Lohnartentabelle 277

lokale Felder in Queries 309
Löschen 156
LOW-DATE 107

M

Makro 106
 PCH 113
Maßnahmenschnellerfassung 221
Matchcode 85
Mehrfachanstellung 96
Mehrfachbeschäftigter 96
Merkmal 116, 118, 206, 212
 Attribute 119
 Entscheidungsbaum 120, 212
 Pflege 118
 Programmoperationen 121
 Rückgabewert 119
 Struktur 119
Merkmal IVWID 45
Metadaten, für Formulare 349
MetaDimension 353
 kundeneigene 355
MetaField 353
MetaFigure 354
MetaNet 350
 anlegen 351
MetaStar 350, 351
 kundeneigener 355
MiniApps 388
minimale Berechtigung 184
Mixed-Mode 403, 415
Mobile Devices 409
modifikationsfrei 182
Modifikationsgruppe 214
Modul-Pool 34
Monatserster 278
Monatsletzter 278

N

Namensaufbereitung 107, 254
Namenskonventionen 244
 Reportklassen 86
NO_DUPLICATE_LANGU 329
NO_INDIRECT_EVALUATION 330

O

Oberflächenstatus 211
OBJEC 102
Object Navigator 193
Objekt-ID 99
Objektyp 46, 99
Operation 241
Organisationsmanagement 98
Organisationschlüssel 158
Organisationsstruktur 85, 162

P

P_ABAP 149
P_APPL 147
P_ORGIN 144
P_ORGINCON 167
P_ORGXX 145
P_ORGXXCON 167
P_PERNR 148
P_TCODE 143
PA70 216
Paartabelle 64
PCL2 92, 279
PDB_PROCESS 81
PE01 242
PE04 246
PERSON_ONLY_ONCE 326
Personalabrechnung 241
 Formulare 347
Personenauswahlzeitraum 84, 88
PFCG 390, 391
Planvariante 99
Plausibilitätsprüfungen 200
PLOG 153
PM01 203
PNP-Modus 96, 97
pnp-sw-found 106
Power-Transaktionen 405
PPCI 225
Primär-Infotyp 44
PRIMARY_INFITY 327
PROC_PERNR_PARTIAL_AUTH 326
PROCESS_LOCKED_RECORD 326
Profilgenerator 139
Programm SAPMPZ02 425
Projektion 90

PROVIDE 89, 327
 Subtyp 90
PROVIDE_FIELD 327
Prüfdatum 157
Prüfkennzeichen 142
Prüfungen 195
Prüfverfahren 156
PSnnnn 40
PSYST 209
Publizieren 408
Pufferung 279
PUT PERNR 81

Q
Query
 erstellen 296
 lokale Felder 309
 Schalter definieren 323
Query Painter 297

R
Regeln 241
REPORT_CLASS 325
Reportattribute 81
Reporteigenschaften 98
Reportklasse 84, 93, 251, 323, 325
 SAP-Defaultklasse 84
Reports 289
 Beispiel-Anbindung eines 433
 RHGR123 295
 Web-enabling von 430
Reportstatistik 276
Repository 193
RGDIR 125
RH-GET-TBDAT 114
RH-SEL-KEYDATE 113
RH-SEL-ONE-OBJID 113
Rolle 139, 389
 Einzelrolle 390
 Länderrolle 390
 Sammelrolle 390
Rollenmenü 139
RP_FILL_WAGE_TYPE_TABLE_EXT
 260
RP_PROVIDE_FROM_FRST 106, 109
RP_PROVIDE_FROM_LAST 109, 257,
 326

RP_READ_ALL_TIME_ITY 111
RP_READ_INFOTYP 110
RP_SET_DATA_INTERVAL 108
RP_UPDATE 111
RP-EDIT-NAME 108
RP-SEL-EIN-AUS-INIT 107
RP-SET-NAME-FORMAT 107
RPTEDT00 347
RPUACG00 179

S

Sammelrolle 139
SAP BW 288
SAP Form Builder 347, 348, 360
SAP GUI für HTML 406, 414
SAP Query 287, 288, 296, 339
SAP Smart Forms 348, 360
SAP-Erweiterungen 129, 165, 198
Schalter, Query 323
 Infotyp-spezifische 323
Schema 241
Schnellerfassung 216
SELECT 104
Selektionsbild 82, 88
 PCH 99
Selektionsdynpro 102
Selektionsfelder 86
Selektionsview 87
Service
 PZLE_01 437
 PZLE_02 437
 PZLE_03 437
 PZLE_04 437
 PZLE_05 437
 PZLE_06 437
 PZLE_07 437
SIAC1 397
Single-Host-Installation 392
Sortierung 85, 280
Sperrkennzeichen 155
Sperrlogik 113
 Funktionsbaustein 113
SPLIT_DATA_REQUIRED 329
SPLIT_DEPENDENT_AF 331
SPLIT_INDEPENDENT_AF 331
Stammdaten 21
 Auswertungen 251

Standardbereich 296
 Statusvektor 100, 164
 Stellenbeschreibung 268
 Stichtag 113, 268
 Stichtagsdatum 251
 STRUC 102
 Struktur
 HRIADMIN 54
 HRIKEY 52
 HRIKEYL 53
 HRIInnnn 55
 PAKEY 36
 pay99_result 95
 payroll 95
 PERNR 83, 88
 Pnnnn 43
 PSHD1 39
 PSHDR 39
 Strukturbedingung 100
 strukturelle Berechtigungsprüfung 137
 Style Editor 423
 Style Sheet Designer 418
 SU24 142
 Subtyp 25
 Suchhilfen 85
 Superuser 151
 symmetrisch 155
 Syntaxprüfung 248

T

Tabelle
 AQLDB 296
 L&W-Tabellen im Überblick 449
 NT1 66
 T500L 71
 T522T 322
 T526 318, 321
 T52IC 337
 T777D 41, 58, 62
 T777I 52
 T777T 52
 T777Z 52
 T778O 47
 T778T 52
 T778V 49
 T77AW 293
 T77WWW_CD 449, 453
 T77WWW_CT 449, 452
 T77WWW_CTP 449, 454
 T77WWW_CTT 449, 452
 T77WWW_LE_EP 442, 450, 454
 T77WWW_LECC 447, 450, 455
 T77WWW_LECCP 448, 450, 455
 T77WWW_LECCT 448, 450, 455
 T77WWW_LEDATA 449, 452
 T77WWW_LESTATUS 448, 450
 T77WWW_MN 449, 451
 T77WWW_MNP 449, 453
 T77WWW_MNT 449, 451
 T77WWW_SDATA 444, 450, 454
 T77WWW_SMAP 446, 449, 454
 T77WWW_SRV 442, 449, 453
 T77WWW_SRVN 449, 453
 TEVEN 64, 66
 Tabelleninfotyp 114, 235, 269
 Tätigkeitsprofile 145
 Template-Generierung 409
 TemSe 379
 Texttabelle 208, 267
 TIME_DEPENDENCY 328
 Toleranzzeit 169
 Transaktion
 HRFORMS 357
 HRUSER 418
 PE50 347
 PFCG 390, 391
 PPIS 291
 PU12 375
 SIAC1 397
 SQ01 296
 SQ02 305
 SQ03 308
 transparente Tabelle HRPnnnn 55
 transparente Tabelle PCL1 67
 TRMAC 105
 TYPE-POOLS 281

U

Umkehrverknüpfung 59
 Umorganisationen 161
 UPDATE 105
 User-Exits des ESS 416

V

Verknüpfung 46, 48, 59
verschiedene Benutzerstammsätze
 166
Vier-Augen-Prinzip 154
Vorschlagswert 195, 224, 253

W

Währungsumrechnung 263
WAP-Service 409
Web Application Builder 396
WebGUI 406
WebStudio 396, 397, 411
 publizieren 408
WGate 392
Wiederholungsinfotypen 114
Wiederholungsstruktur 255

Wurzelobjekt 164

Z

Zeitauswertung 241, 279
 Ergebnisse 277
Zeitbindung 89
Zeitbindungsklasse 28
Zeitereignisse 21, 64
Zeiterfassungsgeräte 64
Zeitlogik 172
Zeitpunkte 253
Zeitwirtschaft 111
 Formulare 347
Zentrale Person 96
zulässige Organisationsschlüssel 159
Zusatzfeld 227
Zuständigkeitszeitraums 171