

# Report Programming in HR



**Release 4.6B**



## Copyright

© Copyright 2000 SAP AG. All rights reserved.

No part of this brochure may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft®, WINDOWS®, NT®, EXCEL®, Word® and SQL Server® are registered trademarks of Microsoft Corporation.

IBM®, DB2®, OS/2®, DB2/6000®, Parallel Sysplex®, MVS/ESA®, RS/6000®, AIX®, S/390®, AS/400®, OS/390®, and OS/400® are registered trademarks of IBM Corporation.

ORACLE® is a registered trademark of ORACLE Corporation, California, USA.

INFORMIX®-OnLine for SAP and Informix® Dynamic Server™ are registered trademarks of Informix Software Incorporated.

UNIX®, X/Open®, OSF/1®, and Motif® are registered trademarks of The Open Group.

HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Laboratory for Computer Science NE43-358, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139.

JAVA® is a registered trademark of Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 USA.

JAVASCRIPT® is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, SAP Logo, mySAP.com, mySAP.com Marketplace, mySAP.com Workplace, mySAP.com Business Scenarios, mySAP.com Application Hosting, WebFlow, R/2, R/3, RIVA, ABAP, SAP Business Workflow, SAP EarlyWatch, SAP ArchiveLink, BAPI, SAPPHIRE, Management Cockpit, SEM, are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

## Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax
	Tip

## Contents

<b>Report Programming in HR</b> .....	<b>6</b>
<b>Logical Database (PA-PAD)</b> .....	<b>7</b>
<b>Activating the Logical Database (PA-PAD)</b> .....	<b>8</b>
<b>Functions (PA-PAD)</b> .....	<b>9</b>
Selection Criteria (PA-PAD).....	10
Selection Screen (PA-PAD).....	12
SAP Naming Conventions for Report Classes (PA-PAD).....	13
<b>Report Structure (PA-PAD)</b> .....	<b>14</b>
<b>Infotype Processing (PA-PAD)</b> .....	<b>16</b>
Processing All Infotype Records (PA-PAD).....	17
Processing a Specific Infotype Record (PA-PAD).....	18
<b>Data Structures (PA-PAD)</b> .....	<b>19</b>
<b>Time Data (PA-PAD)</b> .....	<b>20</b>
<b>Repetitive Structures (PA-PAD)</b> .....	<b>22</b>
<b>Authorization Checks in Reporting (PA-PAD)</b> .....	<b>23</b>
<b>Logical Database (PA-APP)</b> .....	<b>24</b>
<b>Activating the Logical Database (PA-APP)</b> .....	<b>25</b>
<b>Functions (PA-APP)</b> .....	<b>26</b>
<b>Selection Criteria (PA-APP)</b> .....	<b>27</b>
<b>Structure of a Report (PA-APP)</b> .....	<b>28</b>
<b>Infotype Processing (PA-APP)</b> .....	<b>30</b>
Processing all Infotype Records (PA-APP).....	31
Processing a Specific Infotype Record (PA-APP).....	32
<b>Data Structures (PA-APP)</b> .....	<b>33</b>
<b>Authorization Checks in Reporting (PA-APP)</b> .....	<b>34</b>
<b>Views</b> .....	<b>35</b>
Join.....	36
Projection.....	39
Join and Projection.....	41
Time-Dependent Control Tables.....	43
Generalization of the View.....	44
<b>Import/Export Files in HR</b> .....	<b>45</b>
<b>Files PCL1, PCL2, PCL3 and PCL4</b> .....	<b>46</b>
<b>Storing Data in PCLn Files</b> .....	<b>47</b>
xx Key.....	48
<b>PCLn Buffer</b> .....	<b>49</b>
Retroactive Accounting of Payroll Results.....	50
Starting Payroll in the Test Mode.....	51
<b>Cluster Directory</b> .....	<b>54</b>
Function Modules for Selecting Payroll Results.....	55
Function Module: CD_EVALUATION_PERIODS.....	56
Function Module: CD_READ_PREVIOUS.....	58
Function Module: CD_READ_PREVIOUS_ORIGINAL.....	59
Other Modules for the Payroll Cluster.....	60

Explanation of Individual Fields .....	62
Sample Report .....	63
<b>Specific Commands .....</b>	<b>65</b>
<b>Function Modules in HR.....</b>	<b>66</b>
<b>Macro Modules.....</b>	<b>67</b>
<b>Utilities in HR .....</b>	<b>68</b>

## Report Programming in HR

This document is intended for all persons who would like to program their own reports in HR or customize standard ones.

The documentation is not intended to be an introduction to programming. It contains just the special features that form part of programming in HR.

A general knowledge of ABAP programming and HR applications is a prerequisite. This can be acquired by reading the relevant documentation and attending the courses.

This document provides information which enables programmers to become acquainted with the special features of programming in HR.

### Structure of documentation

This documentation is divided into four sections:

The first section gives an introduction to the logical database in HR. The functions of the logical database and the basic structure of the reports which use the logical database are explained. In addition, it describes special evaluation techniques used in Time Management as well as repeat structures and authorization checks.

[Logical database \(PA-PAD\) \[Page 7\]](#)

[Logical database \(PA-APP\) \[Page 24\]](#)

The second section deals with the evaluation of import/export files. How payroll results are read is explained in greater detail.

[Import/export files in HR \[Page 45\]](#)

Specific language elements of HR are presented in the third section. It also describes the function modules and the use of ABAP macros in HR.

[Specific commands \[Page 65\]](#)

The last section presents all tools available in HR.

[Utilities in Human Resources \[Page 68\]](#)

## Logical Database (PA-PAD)

The logical database PNP is provided for evaluation of HR master data and time data. It enables convenient, high-performance evaluation of the transparent tables PAnnnn (nnnn is the infotype number - table PA2011 is an exception.)

[Activating the logical database \(PA-PAD\) \[Page 8\]](#)

[Functions \(PA-PAD\) \[Page 9\]](#)

[Selection criteria \(PA-PAD\) \[Page 10\]](#)

[Selection screen \(PA-PAD\) \[Page 12\]](#)

[Report structure \(PA-PAD\) \[Page 14\]](#)

[Infotype processing \(PA-PAD\) \[Page 16\]](#)

[Data structures \(PA-PAD\) \[Page 19\]](#)

[Time data \(PA-PAD\) \[Page 20\]](#)

[Repetitive structures \(PA-PAD\) \[Page 22\]](#)

[Authorization checks in reporting \(PA-PAD\) \[Page 23\]](#)

[Views \[Page 35\]](#)

---

**Activating the Logical Database (PA-PAD)****Activating the Logical Database (PA-PAD)**

The logical database consists of the database driver SAPDBPNP. If you would like to create your own reports with the logical database, activate the database driver by maintaining the report attributes *Logical database PN* and *Application P*.

## Functions (PA-PAD)

### Data Retrieval

When you run your report, the logical database loads the personnel data for each employee into the main memory and makes it available for processing.

The entire history of each infotype is loaded into the main memory, that is all infotype records from the lowest to highest system date.

The data of the previous personnel number is deleted when you select another personnel number.

**See also:**

[Selection Criteria \(PA-PAD\) \[Page 10\]](#)

[Selection Screen \(PA-PAD\) \[Page 12\]](#)

### Authorization Check

The logical database executes an authorization check for personnel data.

It checks whether the master record of the user who starts the report contains the authorizations for the data that is to be read in the report.

A distinction is made between a person and a data authorization.

The system first checks whether the user has an authorization for the employee in accordance with the criteria of organizational assignment. Employees for which the user has no authorization are not evaluated.

The system then checks whether the user is authorized to process the infotypes of the specified report. A list would be meaningless if the data were not evaluated completely.

**See also:**

[Authorization Checks in Reporting \(PA-PAD\) \[Page 23\]](#)

---

**Selection Criteria (PA-PAD)**

## Selection Criteria (PA-PAD)

The logical database provides several selection criteria. These are divided into the following three groups:

- Person selection
- Sorting the person selection
- Data selection

### Person Selection

The person selections define the set of personnel numbers which are to be evaluated.

The following list contains some of the options for person selection:

- Personnel number
- Person selection period
- Period-determination/payroll area
- Period-determination/payroll period
- Period-determination/payroll year
- Period-determination/indicator
- ... and 30 other selection criteria.

### Person Selection using the Matchcode

You can also use the *Matchcode* function key to select persons. When you select this function, a list of available matchcode IDs is displayed.

### Sorting the Person Selection

Selected employees are processed in ascending order according to personnel number or matchcode. You can use the *Sort* function key to sort employees according to name or organizational assignment criteria.

### Data Selection

- Data selection period

In the data selection field, you can define the period for which infotype records are to be evaluated. Although the entire infotype history is loaded into the main memory during data retrieval, the system only processes those infotype records which are either partly or completely within the data selection period.

- Period determination

You can also define the data selection period using a time period determinator.



**Selection Screen (PA-PAD)**

## Selection Screen (PA-PAD)

Since the required selection criteria depend on the report, you must define the scope of the selection screen.

You can do this by assigning the report to a report class.

The report class defines and determines the required selection criteria and function keys (matchcode, sort, and so on). Reports are assigned to a report class in table *T599B* or *T599W*. The customer maintains the entries in table *T599B*, while table *T599W* contains the SAP assignments. Once reports have been assigned to a class, the customer (table *T599C*) or SAP (table *T599F*) default class settings are valid.

All standard HR reports are already assigned to standard SAP report classes. SAP report classes names comply with the naming convention.

**See also:**

[SAP Naming Conventions for Report Classes \(PA-PAD\) \[Page 13\]](#)



The time evaluation report RPTIME00 is assigned to report class \_\_\_\_0001. In this class, only the *personnel number* and the *matchcodes* function key are permitted as selection criteria (without sorting and time specifications).

The report RPDEDTx0 (Format Payroll Results; x is the country grouping) is assigned to the report class X\_M00002. The selection criteria *Period determination/payroll area*, *Personnel number*, and *Payroll area* are activated here. It is helpful to sort the results in this report (for example, according to personnel area, name), therefore, the function key *Sort* is available next to the function key *Matchcodes*.

For more information on report classes, see Customizing.

***Selections off / Other selections function key***

You can use the *Selections off / Other selections* function key to activate/deactivate specific selection criteria. Selection criteria that have been set cannot be deactivated.

SAP Naming Conventions for Report Classes (PA-PAD)

## SAP Naming Conventions for Report Classes (PA-PAD)

Report class names have eight characters:

S E P D A n n n

These characters have the following meaning:

Character	Meaning	Possible Values	
S	Sort	BLANK	Do not sort
		X	Sort
E	Data selection period =	BLANK	Equality applies
	Person selection period	X	Equality does not apply
P	Payroll area, period and year	BLANK	Line is not displayed
		X	Line is displayed
		M	Line is displayed and payroll area is a required field
D	Data selection period	0	No entry, line is not displayed
		1	Key date; only start date can be entered
		2	Interval; 'start and 'end' dates can be entered
A	Person selection period	Like data selection period	
nnn		Sequential number, begins with 001	



The numeric name range is reserved for customer report classes.

## Report Structure (PA-PAD)

# Report Structure (PA-PAD)

## Example of an HR report

An HR report which uses the logical database has the following basic structure:

```
REPORT RPABAP01.
TABLES: PERNR.
INFOTYPES: 0001.
GET PERNR.
  PROVIDE * FROM P0001 BETWEEN PN-BEGDA AND PN-ENDDA.
  WRITE: / P0001-PERNR,
          P0001-STELL,
          P0001-BEGDA,
          P0001-ENDDA.
ENDPROVIDE.
```

This report evaluates the *Organizational Assignment* infotype records in the specified data selection period.

## Infotype Declaration

All infotypes to be processed in the report are listed in the ABAP `INFOTYPES` keyword.

The database usually contains several records with different validity periods and not just one record for each infotype and personnel number. Infotypes are time-dependent since their data changes over time. For this reason, one structure or work area would not suffice for the provision of infotype data in the main memory.

Therefore, the `INFOTYPES` statement is used to create an internal table for each of the listed infotypes. The structure of this table corresponds to that of the relevant infotype.

## Data Retrieval

Data is retrieved at the `GET PERNR` event. The `GET PERNR` action is executed for all personnel numbers that were selected on the basis of selection screen entries. The event should therefore be viewed as a loop using the selected personnel numbers.

`GET PERNR` fills the internal tables of infotypes that are declared for each employee using the `INFOTYPES` statement.

The internal infotype table is filled with all records existing between the lowest and highest system date. The internal table has the name `Pnnnn`, where `nnnn` is the infotype number.



The header of the internal table `Pnnnn` is undefined after the `GET PERNR` action. In particular, you cannot assume that these headers are reset to their initial values if no records are found for a new personnel number.

For information on processing infotype records, see [Infotype Processing \(PA-PAD\) \[Page 16\]](#).

`PERNR` is a Data Dictionary structure without a database. You must declare this structure in the report using the `TABLES` statement.

### See also:

[Infotype Processing \(PA-PAD\) \[Page 16\]](#)

[Views \[Page 35\]](#)

## Infotype Processing (PA-PAD)

### Infotype Processing (PA-PAD)

We differentiate between processing all infotype records and processing the most recent or the oldest infotype record in the data selection period.

[Processing all infotype records \(PA-PAD\) \[Page 17\]](#)

[Processing a specific infotype record \(PA-PAD\) \[Page 18\]](#)

## Processing All Infotype Records (PA-PAD)

After the `GET PERNR` event, the internal tables of the infotypes contain records and are ready for processing.

Internal tables are generally processed line-by-line using the `LOOP` statement.

The internal tables of infotypes have features which allow special processing.

These tables are defined for specific intervals. In HR, these are time intervals or validity periods.

Processing of infotype records is time-dependent; by this we mean dependent on the data selection period entered on the selection screen. The data of several infotypes can be processed at the same time and made available for a specific partial period.

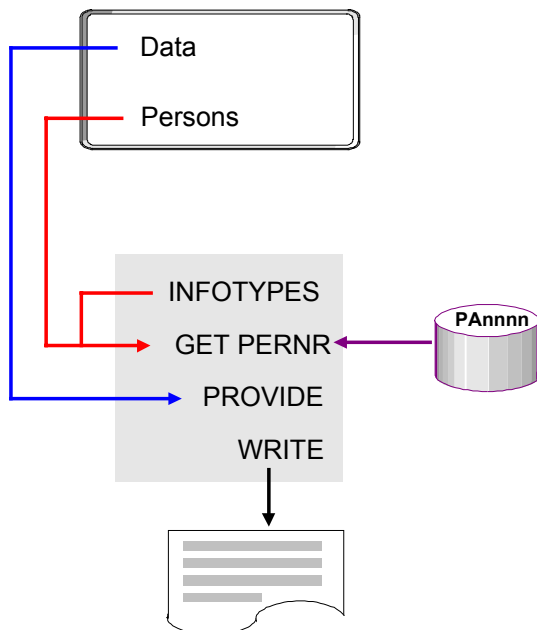
Internal infotype tables are processed with the `PROVIDE` statement.

The syntax is as follows:

```
PROVIDE * FROM Pnnnn BETWEEN PN-BEGDA AND PN-ENDDA.  
  WRITE: / Pnnnn-<field>.  
ENDPROVIDE.
```

`n` stands for the four-digit infotype number. The relationship between the infotype and the data selection period of the selection screen is established using the `PN/BEGDA` and `PN/ENDDA` variables.

In the `PROVIDE` loop, the data of an infotype record is available for processing in the `Pnnnn` structure.



---

**Processing a Specific Infotype Record (PA-PAD)**

## Processing a Specific Infotype Record (PA-PAD)

You often only require the most recent or earliest infotype record, not all infotype records.

In this case, use one of the following statements:

```
RP_PROVIDE_FROM_LAST Pnnnn SPACE PN-BEGDA PN-ENDDA.
```

or

```
RP_PROVIDE_FROM_FIRST Pnnnn SPACE PN-BEGDA PN-ENDDA.
```

These statements make the most recent or earliest record in the PN/BEGDA to PN/ENDDA data selection period available in the structure Pnnnn for infotype nnnn.

If the infotype has subtypes, replace the SPACE parameter by the appropriate subtype number.

When a record has been successfully read, the return code PNP-SW-FOUND = 1 is returned.

**Example report:**

```
REPORT RPDEMO02.  
TABLES: PERNR.  
INFOTYPES: 0001.
```

```
GET PERNR.  
  RP_PROVIDE_FROM_LAST P0001 SPACE PN-BEGDA PN-ENDDA  
  IF PNP-SW-FOUND eq '1'.  
    WRITE: / PERNR-PERNR, P0001-STELL, PN-BEGDA, PN-ENDDA.  
  ELSE.  
    REJECT.  
  ENDIF.
```

The above statements are ABAP macros.

For more information, see [Specific Commands \[Page 65\]](#).

## Data Structures (PA-PAD)

### Structure of HR Master Data and Time Data Tables

HR master data and time data are stored in the transparent tables PAnnnn. In addition to keys (client, personnel number, subtype, object ID, lock indicator, validity period, and sequential number), these tables contain data for the infotype nnnn.

### Structure of Infotypes

The Data Dictionary contains a Pnnnn structure for each infotype nnnn.

The infotype structure Pnnnn corresponds to the table PAnnnn. However, the client, for example, is missing, therefore the infotype number is retained.

The infotype is defined in the Data Dictionary as a structure without a database.

The Pnnnn structure of the infotype is used as the field structure for the infotype entry screen.

When you declare an infotype using the *INFOTYPES* statement, an internal table Pnnnn with the structure Pnnnn is created and all records of the infotype are transferred to this table:

```
DATA BEGIN OF Pnnnn OCCURS 10.  
  INCLUDE STRUCTURE Pnnnn.  
DATA END OF Pnnnn VALID BETWEEN BEGDA AND ENDDA.
```

The infotype records can be processed using the infotype structure when the report is run.

PERNR structure

Event keywords for data retrieval from a logical database have the following syntax:

```
GET <TABLE>.
```

The logical HR database uses the table PERNR. You must declare it in the TABLES statement.

At the GET PERNR event, the PERNR structure contains the data for a personnel number chosen on the basis of selection screen entries.

The PERNR-PERNR field contains the personnel number which is selected for processing.



Only the PERNR-PERNR field should be read from the work area of the PERNR table. The other fields are intended for internal use only.

## Time Data (PA-PAD)

# Time Data (PA-PAD)

## Reading Data

Infotypes 2000 to 2999 are time infotypes. They are stored in tables PA2000 to PA2999. Infotypes are declared with the `INFOTYPE` statement, and data is made available for processing in the internal infotype tables (infotype 2011 is an exception).

You should not load all time infotype records from the earliest to most recent system dates into the main memory. This would quickly lead to a memory overload, especially if a front-end time recording system is connected to your HR system.

This is why time data should be read only for a specific period.

Use the infotype declaration supplement `MODE N` to define that the internal time infotype tables should be declared but not filled at the `GET PERNR` event.

Later you can fill these tables using a statement with selection period parameters.

Use the following report to read time data:

```
REPORT RPABAP05.  
TABLES: PERNR.  
INFOTYPES: 2001 MODE N.  
GET PERNR.  
  RP_READ_ALL_TIME_ITY PN-BEGDA PN-ENDDA.  
  LOOP AT P2001.  
    WRITE: / P2001-ABWTG.  
  ENDLLOOP.
```

An ABAP macro reads the time data. This macro uses the data selection period parameter of the selection screen.

### See also:

[Macro Modules \[Page 67\]](#)

## Processing Data

Due to the time constraint of infotypes, several special features must be taken into account when processing time data. Views of time data are generally not practical.

In time infotypes, data is determined on the basis of the validity period.

When you enter an absence record, the number of days of absence is calculated on the basis of the absence period.

In a view, new partial periods are created without any changes being made to infotype data. This would lead to incorrect results in time infotypes, since this data depends on the validity period.

For example, if, a leave record extends from the middle of January to the middle of February and 20 days of leave are calculated for this period, then a view for the month of February would result in a leave record which extends from the beginning to the middle of February. The number of days of leave would not have changed and the information would be incorrect.

In addition, in master data, the time constraint is a unique characteristic of the infotypes or subtypes. There are no time dependencies between the infotypes and subtypes.

**Time Data (PA-PAD)**

Time data is basically different. Let us assume that an employee becomes sick during vacation. The leave record is then delimited on the first day of the sickness, and the sickness record follows.

Likewise, the system prohibits the entry of a leave record that coincides with a sickness record. The same also applies to overtime during a sickness.

The time dependency of time infotype records covers all infotypes and subtypes and is not limited to dependencies between records of one and the same infotype only.

The time constraint of time infotypes is not an attribute but is defined by the relationships between infotypes.

Moreover, certain time infotype records have specific clock-in/clock out times. Several records may therefore exist for one infotype on a particular day.

Since views require explicit data and this prerequisite is not fulfilled by time infotypes, you should not use joins and projections for time data.

Time infotype tables are processed with the `LOOP` statement since the `PROVIDE` statement limits, and thus changes, the infotype start and end dates to the data selection period.

## Repetitive Structures (PA-PAD)

## Repetitive Structures (PA-PAD)

In many master data infotypes, data is entered in table form. This, for example, allows you to enter up to twenty different wage types and their amounts. The input line for the wage type is available seven times on the input template. By scrolling, you can enter up to twenty wage types.

The structure of the wage type line is stored in the infotype structure P0008, and the individual fields are numbered from one to twenty. This means that each field of the wage type table is defined.

When evaluating repeat structures, you must ensure that all fields are entered. In the case of the *Basic Pay* infotype,  $20 * 5 = 100$  fields are queried.

A loop offers a more streamlined method of evaluation. Here, one line of the repeat structure is evaluated each time the loop is executed.

To use this method of evaluation, define a field string whose structure corresponds to the fields in one line of the repetitive structure.

In this field string, one line of the basic pay wage types is evaluated each time the loop is executed.



```
REPORT RPABAP06.
TABLES:PERNR.
INFOTYPES:    0008.
DATA: BEGIN OF WAGETYPES,
      LGA LIKE P0008-LGA01,
      BET LIKE P0008-BET01,
      ANZ LIKE P0008-ANZ01,
      EIN LIKE P0008-EIN01,
      OPK LIKE P0008-OPK01,
      END OF WAGETYPES.

GET PERNR.

RP_PROVIDE FROM LAST P0008 SPACE PN-BEGDA PN-ENDDA.
DO 20 TIMES VARYING WAGETYPES
FROM P0008-LGA01
NEXT P0008-LGA02.
  IF WAGETYPES-LGA IS INITIAL.
    EXIT.
  ELSE.
    WRITE: / WAGETYPES-LGA, WAGETYPES-BET.
  ENDIF.
ENDDO.
```

Repeat structures are also found in the *Leave Entitlement*, *Cost Distribution*, *Appraisals*, and *Wage Maintenance* infotypes. They are also evaluated in this way.

## Authorization Checks in Reporting (PA-PAD)

Generally, authorization checks in reporting do not differ from those in the transactions. Since data access in reporting is always of the read type, the system checks for a read authorization; this means the authorization group must be **R** or **\***.

In some situations, you may want to use a simplified authorization check when running reports. The object `RPABAP` is required for the check as well as the object `RPORGIN`; if these authorizations are available, a simpler and faster check is performed.

If the report cannot read certain personnel data due to lack of authorization, the data for these persons is not processed at the GET PERNR event. A note appears at the end of the list stating the number of persons who were skipped due to lack of authorization.

---

**Logical Database (PA-APP)**

## Logical Database (PA-APP)

The logical database PAP is provided for evaluation of HR applicant data. It enables convenient, high-performance evaluation of the transparent table PBnnnn (nnnn is the infotype number )

[Activating the logical database \(PA-APP\) \[Page 25\]](#)

[Functions \(PA-APP\) \[Page 26\]](#)

[Selection criteria \(PA-PAD\) \[Page 27\]](#)

[Report structure \(PA-APP\) \[Page 28\]](#)

[Infotype processing \(PA-APP\) \[Page 30\]](#)

[Data structures \(PA-APP\) \[Page 33\]](#)

[Authorization checks in reporting \(PA-APP\) \[Page 34\]](#)

[Views \[Page 35\]](#)

## Activating the Logical Database (PA-APP)

The logical database consists of the database driver SAPDBPAP. If you would like to create your own evaluations with the logical database, activate the database driver by maintaining the report attributes *logical database PNP*.

---

**Functions (PA-APP)**

## Functions (PA-APP)

### Data Retrieval

When you run your report, the logical database loads the personnel data for each applicant into the main memory and makes it available for processing.

The entire history of each infotype is loaded into the main memory, that is all infotype records from the lowest to highest system date.

The data for the previous applicant number is deleted when you select another applicant number.

**See also:**

[Selection Criteria \(PA-PAD\) \[Page 27\]](#)

### Authorization Check

The logical database executes an authorization check for applicant data.

It checks whether the master record of the user who starts the report contains the authorizations for the data that is to be read in the report.

A distinction is made between a person and a data authorization.

The system first checks whether the user has an authorization for the applicant in accordance with the criteria of organizational assignment. Employees for which the user has no authorization are not evaluated.

The system then checks whether the user is authorized to process the infotypes of the specified report. A list would be meaningless if the data were not evaluated completely.

**See also:**

[Authorization Checks in Reporting \(PA-APP\) \[Page 34\]](#)

## Selection Criteria (PA-APP)

The logical database provides several selection criteria. These are divided into the following three groups:

- Applicant selection
- Sorting the applicant selection
- Data selection

When the data is selected, the *Organizational Assignment* (0001), *Applicant Actions* (0002), and *Applications* (4001) infotypes are always read.

When applicants are selected other infotypes, for example, *Personal Data* (0002) in report RPAPL001, are also read, although this depends on the report.

### Applicant Selection

The person selections define the set of applicant numbers which are to be evaluated.

The following list contains some of the options for applicant selection:

- Receipt of application
- Data selection period
- Applicant number
- Advertisement
- Unsolicited application group
- and so on

#### Applicant Selection using the Matchcode

You can use the *Matchcode* function key to select applicants. When you select this function, a list of available matchcode IDs is displayed.

### Sorting the Applicant Selection

Selected applicants are processed in ascending order according to personnel number or matchcode. You can use the *Sort* function key to sort employees according to criteria from the *Organizational Assignment* (0001), *Applicant Actions* (4000) and *Applications* (4001) infotypes.

### Data Selection

In the data selection field, you can define the period for which infotype records are to be evaluated.

## Structure of a Report (PA-APP)

# Structure of a Report (PA-APP)

## Example of a report in HR Recruitment

An HR report which uses the logical database has the following basic structure:

```
REPORT RPABAP01.
TABLES: APPLICANT.
INFOTYPES: 0001.
GET APPLICANT.
  PROVIDE * FROM P0001 BETWEEN PA$BEGDA AND PA$ENDDA.
  WRITE: / P0001-PERNR,
          P0001-ENAME,
          P0001-BEGDA,
          P0001-ENDDA.
ENDPROVIDE.
```

This report evaluates the *Organizational Assignment* infotype records in the specified data selection period.

## Infotype Declaration

All infotypes to be processed in the report are listed in the ABAP `INFOTYPES` keyword.

The database does not normally contain only one record for each infotype and personnel number but several records with different validity periods. Infotypes are time-dependent since their data changes over time. For this reason, one structure or work area would not suffice for the provision of infotype data in the main memory.

Therefore, the `INFOTYPES` statement is used to create an internal table for each of the listed infotypes. The structure of this table corresponds to that of the relevant infotype.

## Data Retrieval

Data is retrieved at the `GET APPLICANT` event. The `GET APPLICANT` action is executed for all personnel numbers that were selected on the basis of selection screen entries. The event should therefore be viewed as a loop via the selected personnel numbers.

`GET APPLICANT` fills the internal tables of infotypes that are declared for each employee using the `INFOTYPES` statement.

The internal infotype table is filled with all records existing between the lowest and highest system date. The internal table has the name `Pnnnn`, where `nnnn` is the infotype number.



Note that the header of the internal tables `Pnnnn` is undefined after the `GET APPLICANT` action. These headers are reset to their initial values if no records are found for a new personnel number.

For information on processing infotype records, see [Infotype Processing \(PA-APP\) \[Page 30\]](#).

`APPLICANT` is a Data Dictionary structure without a database and must be declared in the report under `TABLES`.

### See also:

[Infotype Processing \(PA-APP\) \[Page 30\]](#)

[Views \[Page 35\]](#)

## Infotype Processing (PA-APP)

### Infotype Processing (PA-APP)

We differentiate between processing all infotype records and processing the most recent or the oldest infotype record in the data selection period.

[Processing All Infotype Records \(PA-APP\) \[Page 31\]](#)

[Processing a Specific Infotype Record \(PA-APP\) \[Page 32\]](#)

## Processing all Infotype Records (PA-APP)

After the `GET APPLICANT` event, the internal tables of the infotypes contain records and are ready for processing.

Internal tables are generally processed line-by-line using the `LOOP` statement.

The internal tables of infotypes have features which allow special processing.

These tables are defined for specific intervals. In HR, these are time intervals or validity periods.

Processing of infotype records is time-dependent; by this we mean dependent on the data selection period entered on the selection screen. The data of several infotypes can be processed at the same time and made available for a specific partial period.

Internal infotype tables are processed with the `PROVIDE` statement.

The syntax is as follows:

```
PROVIDE * FROM Pnnnn BETWEEN PA$BEGDA AND PA$ENDDA.  
  WRITE: / Pnnnn-<field>.  
ENDPROVIDE.
```

`nnnn` stands for the four-digit infotype number. The relationship between the infotype and the data selection period of the selection screen is established using the `PA$BEGDA` and `PA$ENDDA` variables.

In the `PROVIDE` loop, the data of an infotype record is available for processing in the `Pnnnn` structure.

See the graphic on [Processing all Infotype Records \(PA-PAD\) \[Page 17\]](#).

---

**Processing a Specific Infotype Record (PA-APP)**

## Processing a Specific Infotype Record (PA-APP)

You often only require the most recent or earliest infotype record, not all infotype records.

In this case, use one of the following statements:

```
PAP_PROVIDE_FROM_LAST Pnnnn SPACE PA$BEGDA PA$ENDDA.
```

or

```
PAP_PROVIDE_FROM_FIRST Pnnnn SPACE PA$BEGDA PA$ENDDA.
```

These statements make the newest or oldest record in the PA\$BEGDA to PA\$ENDDA data selection period available in the structure Pnnnn.

If the infotype has subtypes, replace the SPACE parameter by the appropriate subtype number.

When a record has been successfully read, the return code PAP-SW-FOUND = 1 is transferred.

**Sample report:**

```
REPORT RPDEMO02.  
TABLES: APPLICANT.  
INFOTYPES: 0001.
```

```
GET APPLICANT.  
  PAP_PROVIDE_FROM_FIRST P0001 SPACE PA$BEGDA PA$ENDDA.  
  IF PAP_SW_FOUND eq '1'.  
    WRITE: / APPLICANT-APLNO, P0001-BUKRS, PA$BEGDA, PA$ENDDA.  
  ELSE.  
    REJECT.  
  ENDIF.
```

The above statements are ABAP macros.

For more information, see [Specific Commands \[Page 65\]](#).

## Data Structures (PA-APP)

### Structure of HR Recruitment tables

Recruitment data is stored in the transparent tables Pbnnnn. Apart from the keys (client, applicant number, subtype, object ID, lock indicator, validity period, and sequential number), table PBnnnn only contains the data for the infotype in question.

### Structure of Infotypes

The Data Dictionary contains a Pnnnn structure for each infotype nnnn.

The infotype structure Pnnnn basically corresponds to the table PBnnnn. It contains the infotype number but not the client.

The infotype is defined in the Data Dictionary as a structure without a database.

The Pnnnn structure of the infotype is used as the field structure for the infotype entry screen.

When you declare an infotype using the *INFOTYPES* statement, an internal table Pnnnn with the structure Pnnnn is created and all records of the infotype are transferred to this table:

```
DATA BEGIN OF Pnnnn OCCURS 10.  
  INCLUDE STRUCTURE Pnnnn.  
DATA END OF Pnnnn VALID BETWEEN BEGDA AND ENDDA.
```

The infotype records can be processed using the infotype structure when the report is run.

APPLICANT structure

Event keywords for data retrieval from a logical database have the following syntax:

```
GET <TABLE>.
```

The logical HR database uses the table *APPLICANT*. You must declare it in the *TABLES* statement.

At the *GET APPLICANT* event, the *APPLICANT* structure contains the data for a applicant number chosen on the basis of selection screen entries.

The *APPLICANT-APLNO* field contains the applicant number which is selected for processing.



Only the *APPLICANT-APLNO* field should be read from the work area of the *APPLICANT* table. The other fields are intended for internal use only.

---

**Authorization Checks in Reporting (PA-APP)**

## Authorization Checks in Reporting (PA-APP)

Generally, authorization checks in reporting do not differ from those in the transactions. Since data access in reporting is always of the read type, the system checks for a read authorization; the authorization group must be **R** or **\***.

In some situations, you may want to use a simplified authorization check when running reports. The object `RPABAP` is required for the check as well as the object `RPORGIN`; if these authorizations are available, a simpler and faster check is performed.

If the report cannot read certain applicant data due to lack of authorization, data for these persons is not processed at the `GET APPLICANT` time point. A note appears at the end of the list stating the number of applicants who were skipped due to lack of authorization.

## Views

### Introduction

When evaluating data, we distinguish between the logical and the physical view.

The physical view corresponds to the form in which the infotype data is stored in the HR tables. This data is stored in infotype records with a validity period.

In the logical view, the validity periods of individual fields are determined for several infotype records. For example, for an evaluation, the time period during which an employee worked at a particular job may be important, irrespective of whether a company code, personnel area or cost center change occurred during this time.

Data from several infotypes can also be provided for a specific partial period. When calculating partial payroll periods, it is especially important that data on basic pay, work schedule and cost distribution are provided for the relevant partial period.

These two types of logical views are implemented in the projection and join.

[Join \[Page 36\]](#)

[Projection \[Page 39\]](#)

[Join and Projection \[Page 41\]](#)

[Time-Dependent Control Tables \[Page 43\]](#)

[Generalization of the View \[Page 44\]](#)

## Join

## Join

A join processes records from two or more infotypes. The data from these infotypes is provided for a specific partial period.



We would like to know in which time period an employee worked at which job and at which address he or she resided during this time.

The following address data is available:

January - June	Hamburg
June - December	Munich

The following work center data is available:

January - April	Programmer
May - December	Course instructor

If the address and work center data are provided for specific partial periods, the following cases result:

January - April	Hamburg / programmer
May - June	Hamburg / course instructor
July - December	Munich / Course instructor

The ABAP syntax of this join is as follows:

```
PROVIDE * FROM Pmmmm
         * FROM Pnnnn
         BETWEEN PN-BEGDA AND PN-ENDDA.
```

The partial periods for infotypes Pmmmm and Pnnnn as well as for all other infotypes of the join are defined in the fields BEGDA and ENDDA.

The data of each infotype in the join must be available during the entire validity period of the infotype. The time periods of infotype records may not overlap; therefore, the join may not contain infotypes with time constraint "three".

The time periods of records overlap if an infotype is read without any subtype restrictions. For example, the *Address* infotype has the subtypes *Permanent residence*, *Temporary residence* and *Home address*.

Time periods will ultimately overlap if all addresses are read. Therefore, you must always select a subtype for a join, and this subtype may not have the time constraint "three".

The program code for the above join for work center and address data is as follows:

```
REPORT RPABAP03.
TABLES: PERNR.
INFOTYPES: 0001, 0006.
GET PERNR.
  PROVIDE * FROM P0001
```

```

* FROM P0006 BETWEEN PN-BEGDA AND PN-ENDDA
WHERE P0006-SUBTY eq '1'.
WRITE: / PERNR-PERNR, P0001-STELL,
        P0006-STRAS, P0006-BEGDA, P0006-ENDDA.
ENDPROVIDE.

```

Sometimes no data is available for a particular infotype in the selected partial period. Infotype validity periods may not overlap but gaps are permitted.

For example, gaps can occur when personal data is joined with address data:

Personal data

January 1960 - May 1998	Miller
May 1998 - December 1998	Smith

Address data:

January 1998 - December 1998	Hamburg
------------------------------	---------

A join for personal and address data is presented as follows:

January 1960 - December 1997	Miller
January 1998 - April 1998	Miller / Hamburg
May 1998 - December 1998	Smith / Hamburg

Only personal data is available in the first partial period. Since the record does not provide the required information, the join's function of providing data from all associated infotypes has not been fulfilled.

The variables `Pnnnn_VALID` recognize that only incomplete data is available for a particular partial period.

This variable is formed when the report is run for each `Pnnnn` infotype included in a join.

If data exists in the partial period for the `Pnnnn` infotype, the variable `Pnnnn_VALID` is filled with **X**.

These variables are evaluated as follows:

```

REPORT RPDEMO03.
TABLES: PERNR.
INFOTYPES: 0002,
           0006.
GET PERNR.
  PROVIDE * FROM P0002
          * FROM P0006 BETWEEN PN-BEGDA AND PN-ENDDA
          WHERE P0006-SUBTY = '1'.
  IF P0006_VALID EQ 'X'.
    WRITE: / PERNR-PERNR,
            P0002-BEGDA DD/MM/YYYY,
            P0002-ENDDA DD/MM/YYYY,
            P0002-NACHN,
            P0006-ORT01.
  ENDIF.
ENDPROVIDE.

```

**Join**

A list is generated only if address data is available. The first partial period, for which only personal data is available, is suppressed.

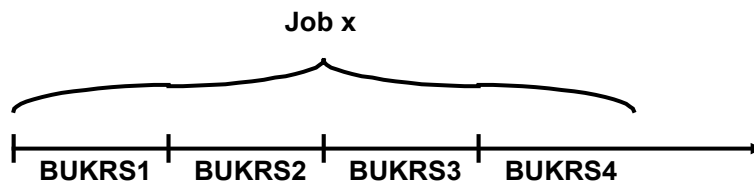
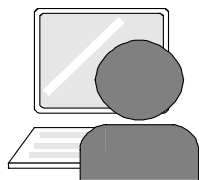
## Projection

All data of an infotype is stored on the database with its period of validity.

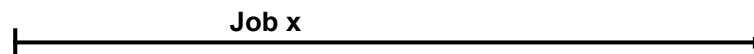
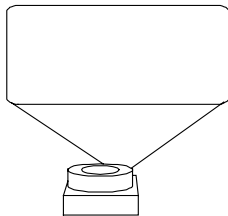
When you change one or more fields of an infotype record, the system creates a new record with a new validity period. The date on which you changed the record is the start date of this new record.

Therefore, the data fields that are not affected by the changes contain the same data over several infotype records and validity periods.

From a logical perspective, these fields are valid in all infotype records until they are changed.



P0001



When seen from a logical perspective, each field of an infotype has its own validity period.

This is illustrated in the following case:

An employee has worked as a programmer for three years in three different personnel areas.

The following organizational assignment data is available:

January 1996 - December 1996: Programmer /personnel area 1

January 1997 - December 1997: Programmer /personnel area 2

January 1998 - December 1998: Programmer /personnel area 3

If you only require the time period during which an employee performs a specific job and not his or her personnel area for an evaluation, the following applies:

January 1996 - December 1998: Programmer

The physical view has three infotype records, the logical view one.

## Projection

To create meaningful evaluations and avoid redundancies, create logical views for infotype records.

Select the infotype fields which are important for the evaluation and disregard the others.

In the above example, the data in the other fields is invalid for the evaluation since it is unknown which personnel area the employee belonged to from 1996 - 1998.

This view of the validity period of a group of infotype fields is known as projection.

The program code for the projection is:

```
PROVIDE <Field_1> <Field_2> <Field_n> FROM Pnnnn  
  BETWEEN PN-BEGDA AND PN-ENDDA.
```

The infotype data for a projection must be available throughout the entire validity period. If the time periods of certain infotype records overlap, the data cannot be clearly assigned to one period.

Therefore, you should not use projections for infotype records with time constraint 'three'. The report for the above projection is:

```
REPORT RPABAP04.  
TABLES: PERNR.  
INFOTYPES: 0001.  
GET PERNR.  
  PROVIDE STELL FROM P0001 BETWEEN PN-BEGDA AND PN-ENDDA.  
  WRITE: / PERNR-PERNR, P0001-STELL, P0001-BEGDA,  
         P0001-ENDDA.  
ENDPROVIDE.
```

The logical validity for the activity period is included in the infotype BEGDA and ENDDA fields.

## Join and Projection

You can combine the two logical views of infotype data, the join and the projection.

We read the data from several infotypes and create new partial periods. We select the infotype fields that are important for the evaluation and combine these partial periods again.

The following example illustrates this.

An employee works as a programmer in the current year and marries in May. Her name does not change.

Organizational assignment:

January - December	Programmer
--------------------	------------

Personal data:

January - April	Donna Debug - single
May - December	Donna Debug - married

When the data from both infotypes is read concurrently, the result is:

January - April	Donna Debug - single / programmer
May - December	Donna Debug - married / programmer

Since we can disregard her marital status in the evaluation, we project on her first and last names:

January - December	Donna Debug / programmer
--------------------	--------------------------

The following report exemplifies the above case:

```
REPORT RPDEMO04.
TABLES: PERNR.
INFOTYPES: 0001,
            0002.
GET PERNR.
  PROVIDE STELL FROM P0001
  NACHN VORNA FROM P0002
  BETWEEN PN-BEGDA AND PN-ENDDA
  IF P0001_VALID = 'X'.
    WRITE: / P0002-NACHN, P0002-VORNA,
            P0001-BEGDA DD/MM/YYYY,
            P0001-ENDDA DD/MM/YYYY,
            P0001-STELL.
  ENDIF.
ENDPROVIDE.
```

This report combines the associated validity periods and provides the data of relevant infotype fields for a specific period.

## Join and Projection



Fields which are not accessed have their initial value in the projection.

Provision of data for a specific partial period is especially important for partial period factoring in payroll.

If an employee's basic pay or the cost distribution changes during the payroll period, you must calculate the salary proportionately for the resulting partial periods.

However, if the payroll administrator of the organizational unit changes, this has no effect on payroll.

By linking a join and a projection, you can read the master data for a specific partial period.

## Time-Dependent Control Tables

Infotype data is generally coded as a key (for example, infotype P0006, address type 1 = permanent residence) to allow fast data entry and space-saving storage. When you process infotypes, the texts or attributes of the keys are read from the relevant control tables.

In many control tables, storage of data is time-dependent and therefore assigned a validity period.

In HR, this applies to the following areas:

- Work schedules
- Pay scale structures
- Wage types
- Wage type valuation
- Bank data
- Positions
- Payee codes

When you read the data for an infotype key from time-dependent control tables, you must determine which record is valid in the specified validity period.

If you use a transaction to process an infotype, the system reads the table record which is valid on the start date.

---

**Generalization of the View**

## Generalization of the View

You can use the logical view to edit and output data according to user specifications.

The special feature of HR views is the time dependency of the data. Personnel data is almost always related to specific validity periods. A HR view provides data for specific time intervals.

In general terms, a HR view is a logical perspective of interval-dependent internal tables.

**See also:**

[Processing All Infotype Records \(PA-PAD\) \[Page 17\]](#)

[Processing All Infotype Records \(PA-APP\) \[Page 31\]](#)

[Processing a Specific Infotype Record \(PA-PAD\) \[Page 18\]](#)

[Processing a Specific Infotype Record \(PA-APP\) \[Page 32\]](#)

## Import/Export Files in HR

The following sections describe the purpose of files PCL1 and PCL2 and explains how to access them.

[Files PCL1, PCL2, PCL3 and PCL4 \[Page 46\]](#)

[Storing Data in PCLn Files \[Page 47\]](#)

[PCLn Buffer \[Page 49\]](#)

[Cluster Directory Manager \[Page 54\]](#)

---

**Files PCL1, PCL2, PCL3 and PCL4**

## Files PCL1, PCL2, PCL3 and PCL4

### Which information is stored in the files?

File PCL1 is the basis for the HR work area data. It contains information from the time data recording, for example, incentive wage time tickets or infotype supplement texts.

File PCL2 contains derived information, for example, payroll results. It also contains all generated payroll schemas.

File PCL3 contains applicant data.

File PCL4 contains the change documents for HR master data and recruitment.

The structure of PCLn files corresponds to that of the INDX file which you may be familiar with from other applications. The structure of all PCLn files (n = 1, 2, 3, and 4) is identical.

### Structure of Files

Like in almost all SAP files, the key element with the highest priority is the client; data within a client is grouped according to basic relations (field *PCLn-RELID*).

The type of basic relation is known as a cluster and characterizes the stored data according to the type, for example, cluster RX contains the payroll result for country X (from table T500L) and cluster TE contains the trip costs data.

Depending on the cluster, the structure of *PCLn-SRTFD* is defined in a field string [xx-KEY \[Page 48\]](#), which is defined in an include *RPCnxy0*.

### Naming conventions

n	=	1, 2, 3, or 4 (for PCL1, PCL2, PCL3, or PCL4)
xx	=	for the cluster
y	=	0 for international clusters
y	=	country grouping according to T500L for national clusters

The personnel number is usually the first component of [xx-KEY \[Page 48\]](#).

### Importing and Exporting Data

The import/export files PCLn are managed with the ABAP/4® commands `IMPORT` and `EXPORT`. These commands store objects such as fields, field strings, or internal tables on the database, or read these from the database. Data is read from and written to the database using a unique key([xx-Key \[Page 48\]](#)).

Please note that the RMAC macros *RP-IMP-Cn-xx* and *RP-EXP-Cn-xx* are provided for importing and exporting data. Only these macros should be used.

See also [Macro Modules \[Page 67\]](#)

## Storing Data in PCLn Files

Data from the different HR application areas is stored in data clusters in PCLn files (n = 1, 2, 3, or 4).

This collection of data objects can consist of:

- Fields used within reports
- Field strings
- Internal tables

The structure of the PCLn files provides a framework for the individual application areas.

Each application area must have a two-character cluster name (relation ID). It must also have a key structure; 40 bytes of the *SRTFD* field are available for this structure.

When a record is exported to the PCLn file, the cluster ID is written to the *RELID* field and the key value to the *SRTFD* field.

Naming convention for includes when defining clusters:

RPCnxy0	n	=	1, 2, 3 or 4 (for PCL1, PCL2, PCL3, PCL4)
	xx	=	cluster ID
	y	=	country indicator

### Description of Cluster Data using Cluster RX as an Example

The data definition is stored in the include `RPC2RX00` in accordance with the above naming conventions.

Structure of cluster key:

```
Data: BEGIN OF RX-KEY.
      INCLUDE STRUCTURE PC200.
      DATA: END OF RX-KEY.
```

The DDIC structure `PC200` contains the fields *PERNR* (personnel number) and *SEQNO* (sequential number).

The data definition of the cluster also contains other internal tables.

For a list of available data clusters, refer to the domain description in the Data Dictionary.

**xx Key****xx Key**

The xx key name is dependent on the cluster.

The RX KEY is used for all Rx and Xx clusters. In all other cases, the name of the xx key corresponds to that of the cluster.



Cluster	xx Key
RA	RX-KEY
B1	B1-KEY
G3	G3-KEY
XA	RX-KEY

## PCLn Buffer

To keep the number of database accesses to a minimum, import and export data is stored in the main memory buffer. Buffer management routines ensure that exported data can be stored in the PCLn files.

The following two examples illustrate which problems can occur without a buffer.



[Retroactive accounting of payroll results \[Page 50\]](#)

[Starting payroll in the test mode \[Page 51\]](#)

---

**Retroactive Accounting of Payroll Results**

## Retroactive Accounting of Payroll Results

In February 1998, a retroactive accounting run is executed for January.

FOR PERIOD 199801 IN PERIOD 199802

The payroll results for January are recalculated and then written directly to the database.

Result:

The database now contains the results of the following payroll periods.

FOR-PERIOD 199801 IN-PERIOD 199802

FOR-PERIOD 199801 IN-PERIOD 199801

Payroll is then run for February.

FOR-PERIOD 199802 IN-PERIOD 199802

If problems should arise during the payroll run for this period, the February record is not stored on the database.

Result:

The current January record on the database is:

FOR-PERIOD 199801 IN-PERIOD 199802

This problem does not arise if you use the buffer since all data of a transaction is always updated collectively. In the above example, the recalculated January result would be stored in the buffer and, if the payroll run for February were terminated prematurely, the database would not be updated.

The current January record on the database would thus be:

FOR-PERIOD 199801 IN-PERIOD 199802

## Starting Payroll in the Test Mode

In a test run, the database is not updated. Since the payroll results from the previous period are used as the basis for calculating the results of the following period, the results of the actual payroll run would differ from those of the test run, if this test run were executed over several periods.

The use of the buffer enables trouble-free access to the required results for the previous period.

What is required for exporting/importing data to/from the PCLn files using the buffer?

- The following includes contain the data definition for the buffer. They must be included in the report that writes the data to or reads the data from the database.

RPPPXD00

RPPPXD10



Include RPPPXD10 must be in the `common part 'BUFFER'` .

Include RPPPXM00, which contains the buffer management routines, is also required.

The macros for importing and exporting data must comply with the following naming convention:

### Naming Convention for EXPORT/ IMPORT Macros:

RP-aaa-Cn-xy

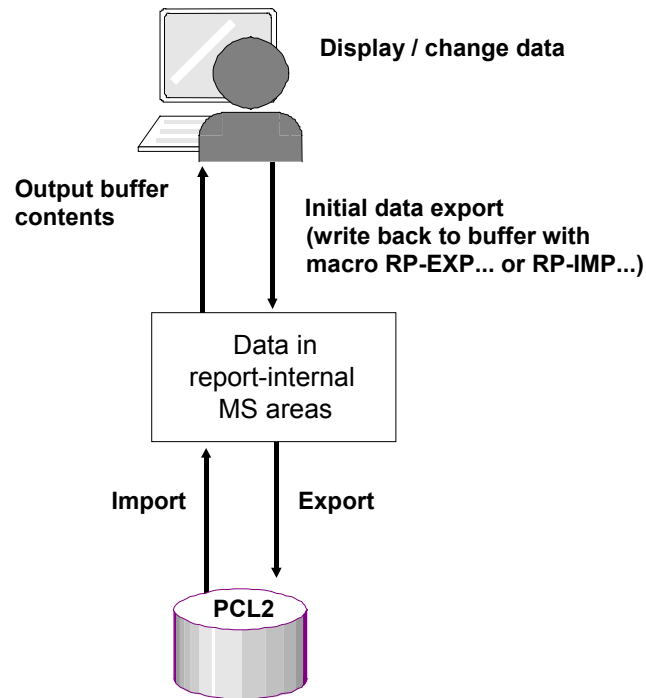
where aaa = IMP / EXP, n=1 for PCL1, 2 for PCL2, 3 for PCL3, 4 or PCL4  
and xy = cluster name.

This guarantees consistency between the export and import of data and also ensures that all exported objects are imported again.

### Export Using the Data Buffer

When macros are used for exporting, records are written to a main memory buffer and not directly to the database. When the program run has been completed, the records in the buffer are stored in the appropriate PCLn database.

## Starting Payroll in the Test Mode



## Import Using the Data Buffer

When the macros are used to import data, the data records are not read directly from file PCLn. Instead, the system checks the buffer directory to see whether the main memory already contains a record with the same key. If this is not the case, the record is read from PCLn to the buffer and then retrieved from the buffer for the report.

If the import is successful, the return code `RP-IMP-xy-SUBRC = 0` is set. When data is read from the buffer, the system carries out a check for cluster authorization. Standard import programs follow the naming convention `RPCLSTxy` (`xy` = cluster name).



```
report rpttdmg.
```

```
tables:
```

```
  pernr,
```

```
  pcl1,
```

```
  pcl2.
```

```
include rppxd00. "buffer definitions
```

```
  data: begin of common part 'BUFFER'.
```

```
  include rppxd10.          "PCLx buffer
```

```
  data: end of common part.
```

```
  data: begin of common part 'CLUSTER_DIRECTORY'.
```

Starting Payroll in the Test Mode

```

include rpc2cd00.      "cluster directory definitions
data: end of common part.

include rpc2rdd0.

get pernr.

rp-init-buffer.      "reset buffer
cd-key-pernr = pernr-pernr.
rp-imp-c2-cd.        "read cluster CD from
buffer/DB
perform cd_manager using ... .
* alternative: call function rp_evaluation_periods...
...
rx-key-pernr = pernr-pernr.
rx-key-seqno = rgdir-seqnr.
rp-imp-c2-rd.        "read cluster RD from
buffer/DB
...
rp-exp-c2-rd.        "update cluster RD in buffer
...
perform prepare_update using 'V'. "update database (DB)
*-----
* Subroutines CD manager and Cluster buffer
*-----
include rpcmgr00.    "Cluster Directory Manager
include rpppxm00.    "module pcl1(2)-buffer

```

## Cluster Directory

### Finding Payroll Results for a Specific Query

Payroll results are stored in cluster Rx of the PCL2.

The cluster key is non-mnemonic. It contains the *PERNR* (personnel number) and *SEQNO* (sequential number) fields.

The internal table RGDIR contains a directory entry for each payroll result. This entry is a sequential number (RGDIR-SEQNR) which uniquely identifies the payroll result.

Payroll results can only be imported if the payroll cluster key contains the personnel number and sequential number.

Before you can import a payroll record, you must select the entry in the RGDIR on the basis of existing data such as for-period, for-payroll area, for-payroll category, in-period, in-payroll area, in-payroll category, and so on, in order to determine the sequential number.

You will probably always have the same queries when importing payroll records. For example, "Which payroll results (original and retroactively accounted records) were written for a specific payroll run (defined by IN payroll category, IN payroll area, IN period)?"

There are standard modules that can be used. It is advantageous to use the standard modules rather than self-programmed solutions because no program modifications will be required if the payroll directory changes. The modules are described in the following section:

[Function Modules for Selecting Payroll Results \[Page 55\]](#)

## Function Modules for Selecting Payroll Results

The employee's payroll directory is always transferred to the function modules using the table RGDIR.

The modules then transfer the payroll records which satisfy the specified selection criteria using a table whose type corresponds to that of the RGDIR but which has a different name. The selection parameters differ according to the function of the module. For more information, read the module documentation.

All module names begin with 'CD\_'.

[Function Module: CD\\_EVALUATION\\_PERIODS \[Page 56\]](#)

[Function Module: CD\\_READ\\_PREVIOUS \[Page 58\]](#)

[Function Module: CD\\_READ\\_PREVIOUS\\_ORIGINAL \[Page 59\]](#)

[Other Modules for the Payroll Cluster \[Page 60\]](#)

[Sample Report \[Page 63\]](#)

## Function Module: CD\_EVALUATION\_PERIODS

**Function Module: CD\_EVALUATION\_PERIODS**

This module transfers the payroll results to a payroll run as 'A' records (current). It also transfers the accompanying 'P' records (previous).

This is the module most frequently used in evaluation programs.



Table contents before the function module is accessed:

SEQNR	FPPER	FPBEG	FPEND	INPER	IPBEG	IPEND	BONDT	PAYTY	PAYID	INPTY	INPID
00001	01.199 6	01.01.9 6	01.15.9 6	01.199 6	01.01.9 6	01.15.9 6					
00002	01.199 6	01.01.9 6	01.15.9 6		01.16.9 6	01.16.9 6				B	0
00003		01.16.9 6	01.16.9 6		01.16.9 6	01.16.9 6	01.16.9 6	B	0	B	0
00004		01.17.9 6	01.17.9 6		01.17.9 6	01.17.9 6	01.17.9 6	A	0	A	0
00005		01.17.9 6	01.17.9 6		01.17.9 6	01.17.9 6	01.17.9 6	A	1	A	1
00006	02.199 6	01.16.9 6	01.31.9 6	02.199 6	01.16.9 6	01.31.9 6					
00007	02.199 6	01.16.9 6	01.31.9 6	03.199 6	02.01.9 6	01.15.9 6					
00008	03.199 6	01.16.9 6	01.31.9 6	03.199 6	01.01.9 6	02.15.9 6					

The following parameters are transferred:

- BONUS\_DATE = '00000000'
- INPER\_MODIF = '02'
- INPER = '199803'
- PAYTY = ' '
- PAYID = ' '

Result:

SEQNR	FPPER	FPBEG	FPEND	INPER	IPBEG	IPEND	BONDT	PAYTY	INPTY	INPID	SRTZA
00006	02.199 6	01.16.9 6	01.31.9 6	02.199 6	01.16.9 6	01.31.9 6					P
00007	02.199 6	01.16.9 6	01.31.9 6	03.199 6	02.01.9 6	02.15.9 6					A
00008	03.199 6	01.16.9 6	01.31.9 6	03.199 6	02.01.9 6	02.15.9 6					A

[Explanation of individual fields \[Page 62\]](#)

Function Module: CD\_READ\_PREVIOUS

## Function Module: CD\_READ\_PREVIOUS

This module transfers a previous payroll record for a payroll record; this is the newest record for the payroll period (or daily payroll run) which was written before the transferred payroll record and contains the same FOR data as the transferring record.



Table contents before the function module is accessed:

SEQNR	FPPER	FPBEG	FPEND	INPER	IPBEG	IPEND	BONDT	PAYTY	PAYID
00001	01.1996	01.01.96	01.15.96	01.1996	01.01.96	01.15.96			
00002	01.1996	01.01.96	01.15.96		01.16.96	01.16.96			
00003		01.16.96	01.16.96		01.16.96	01.16.96	01.16.96	B	0
00004		01.17.96	01.17.96		01.17.96	01.17.96	01.17.96	A	0
00005		01.17.96	01.17.96		01.17.96	01.17.96	01.17.96	A	1
00006	02.1996	01.16.96	01.31.96	02.1996	01.16.96	01.31.96			
00007	02.1996	01.16.96	01.31.96	03.1996	02.01.96	02.15.96			
00008	03.1996	01.16.96	01.31.96	03.1996	02.01.96	02.15.96			

The following parameters are transferred:

- Record with SEQNR '00007'

Result:

SEQNR	FPPER	FPBEG	FPEND	INPER	IPBEG	IPEND	BONDT	PAYTY	SRTZA
00006	02.1996	01.16.96	01.31.96	02.1996	01.16.96	01.31.96			P

[Explanation of individual fields \[Page 62\]](#)

## Function Module: CD\_READ\_PREVIOUS\_ORIGINAL

This module reads the previous original payroll result.



Table contents before the function module is accessed:

SEQNR	FPPER	FPBEG	FPEND	INPER	IPBEG	IIPEND	BONDT	PAYTY	PAYID
00001	01.1996	01.01.96	01.15.96	01.1996	01.01.96	01.15.96			
00002	01.1996	01.01.96	01.15.96		01.16.96	01.16.96			
00003		01.16.96	01.16.96		01.16.96	01.16.96	01.16.96	B	0
00004		01.17.96	01.17.96		01.17.96	01.17.96	01.17.96	A	0
00005		01.17.96	01.17.96		01.17.96	01.17.96	01.17.96	A	1
00006	02.1996	01.16.96	01.31.96	02.1996	01.16.96	01.31.96			
00007	02.1996	01.16.96	01.31.96	03.1996	02.01.96	02.15.96			
00008	03.1996	01.16.96	01.31.96	03.1996	02.01.96	02.15.96			

The following parameters are transferred:

- Record with SEQNR '00008'

Result:

SEQNR	FPPER	FPBEG	FPEND	INPER	IPBEG	IPEND	BONDT	PAYTY	SRTZA
00006	02.1996	01.16.96	01.31.96	02.1996	01.16.96	01.31.96			P

[Explanation of individual fields \[Page 62\]](#)

## Other Modules for the Payroll Cluster

## Other Modules for the Payroll Cluster

Modules which derive information from the payroll cluster are available in addition to the modules for payroll result selection.

### 1. CD\_RETROCALC\_PERIOD

This module differentiates between original payroll records and retroactive accounting records.



Table contents before the function module is accessed:

SEQNR	FPPER	FPBEG	FPEND	INPER	IPBEG	IPEND	BONDT	PAYTY	PAYID
00001	01.1996	01.01.96	01.15.96	01.1996	01.01.96	01.15.96			
00002	01.1996	01.01.96	01.15.96		01.16.96	01.16.96			
00003		01.16.96	01.16.96		01.16.96	01.16.96	01.16.96	B	0
00004		01.17.96	01.17.96		01.17.96	01.17.96	01.17.96	A	0
00005		01.17.96	01.17.96		01.17.96	01.17.96	01.17.96	A	1
00006	02.1996	01.16.96	01.31.96	02.1996	01.16.96	01.31.96			
00007	02.1996	01.16.96	01.31.96	03.1996	02.01.96	02.15.96			
00008	03.1996	01.16.96	01.31.96	03.1996	02.01.96	02.15.96			

The following parameters are transferred:

- Record with SEQNR '00008'

Result:

- CALCD = ' '

[Explanation of individual fields \[Page 62\]](#)

### 2. CD\_PAYROLL\_UNTIL

This module reads the RGDIR for the date up to which the regular payroll run was executed for an employee.

### 3. CD\_HIGHEST\_PAYDT

This module reads the most recent check date for an employee from the RGDIR.

### 4. CD\_GET\_INFO

This module provides information (most recent check date, accounted to date) for a particular personnel number.



---

**Explanation of Individual Fields**

## Explanation of Individual Fields

### For-Information

The FPPER, FPBEG, FPEND, BONDT, PAYTY, PAYID, ABKRS, PERMO, PAYDT, JUPER fields contain information on the period for which payroll is run.

### In-Information

The INPER, IPEND, INPTY, INPID, IABKRS, IPERM fields contain information on the period in which payroll is run.

### SEQNR

The field is used as a key to uniquely identify the payroll record.

This field also defines the sequence of payroll results (history).

### Control Indicator (SRTZA)

Control indicator	Meaning
a	Current
p	Previous
o	Old



For more information, see the online documentation for the individual function modules.

## Sample Report

REPORT RPTTMWBS.

DATA: RGDIR LIKE PC261 OCCURS 0 WITH HEADER LINE.  
DATA: EVPDIR LIKE RGDIR OCCURS 0 WITH HEADER LINE.  
DATA: PREVIOUS\_RESULTS LIKE RGDIR OCCURS 0 WITH HEADER LINE.  
DATA: CALCD TYPE C.  
DATA: IN\_ENTRY LIKE PC261.  
DATA: OUT\_ENTRY LIKE PC261.

INCLUDE RPCCCD09.

```
CALL FUNCTION 'CU_READ_RGDIR'
  EXPORTING
    PERSNR           = '00021218'
    TABLES IN_RGDIR = RGDIR
  EXCEPTIONS
    NO_RECORD_FOUND = 1
    OTHERS           = 2.
* Read RGDIR

CALL FUNCTION 'CD_EVALUATION_PERIODS'
  EXPORTING
    BONUS_DATE      = '00000000'
    INPER_MODIF     = '02'
    INPER           = '199603'
    PAY_TYPE        = CD_C-REGULAR
    PAY_IDENT       = ' '
  TABLES
    RGDIR           = RGDIR
    EVPDIR          = EVPDIR
*   IABKRS         =
  EXCEPTIONS
    NO_RECORD_FOUND = 1
    OTHERS          = 2.
* output:
* 00006
* 00007
* 00008

* Read regular payroll results for January
* A results (original result plus retroactive calculations)
* and P results

LOOP AT EVPDIR WHERE SRTZA = CD_C-ACTUAL.

* Only current results (00007 and 00008)

CALL FUNCTION 'CD_RETROCALC_PERIOD'
  EXPORTING
    ENTRY = EVPDIR
  IMPORTING
    CALCD = CALCD
  EXCEPTIONS
    OTHERS = 1.
* Determine, whether original result
  CHECK CALCD = ' '.
* Special processing: Only the original period
```

**Sample Report**

```
*   March is processed (seqnr 00008).
      IN_ENTRY = EVPDIR.

CALL FUNCTION 'CD_READ_PREVIOUS_ORIGINAL'
  EXPORTING
    IN_RECORD = IN_ENTRY
  IMPORTING
    OUT_RECORD = OUT_ENTRY
  TABLES
    RGDIR      = RGDIR
  EXCEPTIONS
    OTHERS     = 1.

*   out_entry now contains the previous results
*   Input 00008 ----> Output 00006
ENDLOOP.

LOOP AT EVPDIR WHERE SRTZA = CD_C-ACTUAL.
  IN_ENTRY = EVPDIR.

CALL FUNCTION 'CD_READ_PREVIOUS'
  EXPORTING
    IN_RECORD = IN_ENTRY
  TABLES
    RGDIR      = RGDIR
    OUT_RGDIR  = PREVIOUS_RESULTS
  EXCEPTIONS
    NO_RECORD_FOUND = 1
    OTHERS       = 2.

* Input 00007 ---> 00006
* Input 00008 ---> no record found

* Output structure is a table, since there can be
* several previous results: for example, if legal person
* changes, and is retroactively deleted
ENDLOOP.
```

## Specific Commands

The following sections describe the different specific commands in HR.

[Function modules in HR \[Page 66\]](#)

[Macro modules \[Page 67\]](#)

## Function Modules in HR

Function modules are program modules which have a defined interface and allow type testing of parameters.

They are managed with transaction SE37 and combined to function groups according to relevant criteria. You can access this transaction under *Tools* → *ABAP Workbench* → *Function Builder*.

The HR function groups use the naming convention RPxx or HRxx where xx is an identifier of your choice.

You can use the *SHOW FUNCTION* \* editor command to branch from report processing to function module display.

## Macro Modules

### Definition

An module that can be called within an ABAP program.

### Use

Like subprograms and function modules, macro modules are a means of presenting programs in modular form. Macro modules (macros) are used often in the *Human Resources* application component (HR).

### Defining and Calling Modules

Two options are provided:

- Macros can be defined in reports or includes using the ABAP command DEFINE. A macro can be used within a report or within an include. If a macro is used in a report, and the macro is defined in an include with the DEFINE command, the include must be integrated.



Macros have the following advantages:

If a macro is changed, each report using this macro is automatically regenerated when it is executed.

- Macros can also be defined as RMAC macros. The source code of these modules is stored in the function section of the control table TRMAC (Macros in ABAP Programs). The coding is grouped under a specific name in the table key. According to conventions, the first two letters of the name must stand for the application. The rest of the name is freely definable.



Customer-specific RMAC modules should begin with a special character. The macros defined in the control table TRMAC can be used by all reports.



When you change a RMAC macro in the table TRMAC, the reports that use this macro are not regenerated automatically. You must regenerate them manually.

The following section includes a list of [programming utilities for the logical databases PNP and PAP \[Ext.\]](#).

## Utilities in HR

## Utilities in HR

The following utilities are available.

### General Utilities

Report	Meaning
RPUACG00	Code generation / authorization check
RPUAUD00	Infotype auditing

### Programming Utilities

Report	Meaning
RPINCL10	String search in reports

### Cluster Utilities

Report	Meaning
RPCLSTyy	Display cluster for PCLx (yy = RELID)
RPUPxD00	Delete cluster for PCLx (individual data records)
RPUPxD10	Delete cluster for PCLx (several data records)