

SAP Query (BC-SRV-QUE)



Release 4.6C



Copyright

© Copyright 2001 SAP AG. All rights reserved.

No part of this publication 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®, PowerPoint® 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.

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, Massachusetts Institute of Technology.

JAVA® is a registered trademark of Sun Microsystems, Inc.

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

SAP, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com 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

Inhalt

SAP Query (BC-SRV-QUE)	10
InfoSet Query	11
Maintaining Queries	13
Starting InfoSet Query	16
InfoSet Display	18
Editing the InfoSet Display	21
Selection Display	23
Output List Display	25
Formatting the Output	27
Reusing Queries	30
Creating New Queries	31
Loading Queries	33
Deleting Queries	34
Saving Queries	35
Saving a Query	37
Change InfoSet	38
Settings	39
Logging	42
Activating Logging	43
Analyzing Log Data	44
Deleting Log Data	45
Calling the InfoSet Query	46
Rights to Access InfoSet Query	47
Type of Reporting	48
Modules for Calling the InfoSet Query	49
Including InfoSet Query in Roles	51
HR in InfoSet Query	52
Switch Object Selection On/Off	54
Overview of Functions	55
Selection	58
Make Selection	60
Edit Selected Set of Objects	62
Restrict the Reporting Set	63
Set Operations	64
Execute Set Operations	65
Output	66
Output Data for Selected Objects	68
Process Selection Result in General Reporting	69
HR InfoSets for InfoSet Query	70
SAP Query	72
List of Components	73
Groups of Query Users	74
Maintaining Queries	75
Maintaining User Groups	76
Maintaining InfoSets	77
Query Areas	78

Transports	79
Language Comparison	80
SAP Query Authorizations	81
Release Notes: Changes to SAP Query	83
Application	84
Functions for Managing Queries	85
Choosing a Query for Execution or Maintenance	86
Displaying Directories	88
User Group Directory	89
Directory of Queries	90
InfoSet Directory	92
Managing Queries	93
Displaying Queries	94
Copying Queries	98
Renaming Queries	99
Deleting Queries	100
Comparing Queries	101
Executing Queries	102
Executing Queries Online	103
Executing Queries in the Background	107
Improving Response Times	108
Interactive List Display Functions	109
Saving and Redisplaying Lists	113
Calling Other Reports	116
Interactive Functions for Further List Processing	117
Common Principles when Passing Lists for Further Processing	118
Download to File	119
Table Calculation	120
Graphics	121
Word Processing	123
ABC Analysis	124
EIS	128
Private File	129
Additional Function Pool	130
Direct Interaction	132
Creating and Changing Queries	133
Creating Basic Lists with the Query Painter	134
Creating Queries	136
Selecting Fields	138
Assigning Title, Format, and Notes	139
Selecting Fields for Processing	142
Assigning Short Names	144
Defining Local Fields	145
Extending the Selection Criteria	149
Defining Basic Lists	150
Defining a Single-Line Basic List	151
Defining the Line Structure	152
Defining Output Options for Each Line	155

Defining Output Options for Each Field.....	156
Changing Headers	158
Graphics	161
Sorting	162
Output Options for Control Levels	164
Defining Multiple Line Basic Lists	168
Sorting and Subtotals	170
Changing Control Level Texts	174
Totals Lists	176
Output Positions of Fields	178
Defining Statistics	180
Generating Statistics	181
Sorting Statistics	183
Defining Headers in Statistics	188
Error Analysis with Conversion Errors.....	189
Defining Ranked Lists.....	190
What are Ranked Lists?	191
Creating Ranked Lists	192
Changing Queries	194
Maintaining Queries in the Human Resources Application (HR).....	195
Line Groups	196
Evaluations According to a Company's Organizational Structure	198
System Administration	200
Managing InfoSets	201
Selecting InfoSets for Processing.....	202
Displaying the InfoSet Directory	203
Directory of Queries for an InfoSet	204
Query Directory Functions	205
Saved Lists Directory.....	206
Maintaining InfoSets	207
Displaying InfoSets	208
Copying InfoSets	209
Renaming InfoSets	210
Deleting InfoSets	211
Saving InfoSets.....	212
Generating InfoSets	213
Assigning InfoSets to User Groups	215
Checking InfoSets.....	216
Adjusting InfoSets.....	217
Creating and Changing InfoSets.....	218
Creating InfoSets	219
Assigning Data Sources	220
InfoSet Display.....	223
Text Fields	225
Definition of Field Groups	226
Assigning Fields to a Field Group.....	228
Deleting Field Groups/Fields from Field Groups	229
Displaying a List of Assigned Fields	230

Obtaining Additional Information.....	231
Assigning Additional Tables	234
Assigning Table Fields to a Field Group	238
Displaying a List of Assigned Additional Tables.....	239
Deleting the Assignment of Additional Tables.....	240
Creating Additional Fields.....	241
Assigning Additional Fields to a Field Group	244
Displaying a List of Existing Additional Fields	245
Deleting Additional Fields.....	246
Creating Additional Structures.....	247
Assigning Structure Fields to Field Groups.....	248
Deleting the Assignment of Additional Structures	249
Creating Extra Selection Fields	250
Creating Parameters.....	252
Creating Selection Criteria.....	254
Displaying a List of Existing Parameters	256
Further Code.....	257
DATA Code.....	259
START Code	260
END Code.....	261
GET/GET LATE Code	262
TOP-OF-PAGE Code	263
Application-specific Enhancements	264
InfoSets in the HR Application	266
HR Logical Databases	270
Creating InfoSets without an Underlying Logical Database.....	271
Graphical Table Join Definition.....	272
Definition of a Table-Join without Graphics	274
Direct Read	277
Sequential Datasets.....	278
Retrieving Data with Programs	279
Special Features	282
Code for Record Processing.....	284
Functions for Managing User Groups	285
Displaying User Group Directories	286
Managing User Groups.....	287
Displaying User Groups.....	288
Creating User Groups.....	289
Assigning Users and InfoSets	290
Assigning Users to User Groups	291
Assigning InfoSets.....	292
Copying User Groups	293
Renaming User Groups	294
Deleting User Groups	295
Changing User Groups.....	296
Query Areas	297
Maintaining the Additional Function Pool.....	301
Transporting Query Objects	303
Transporting Global Area Objects	304

Transporting Standard Area Objects	306
General Transport Description	307
Generating Transport Datasets	310
Transport Types	311
Transporting a User Group.....	313
Transporting InfoSets	314
Transporting a Query.....	315
Reading Transport Datasets.....	316
Managing Transport Datasets	318
Transporting Objects Between Query Areas	319
Language Comparison.....	320
General Language Comparison Procedure	321
Language Comparison of an InfoSet	325
Language Comparison of a Query.....	327
Language Comparison for User Groups.....	330
Appendix	331
Overview of SAP Query Functions	332
Queries.....	333
Enhancement SQUE0001: Private File.....	334
Example	340
Access Optimization in Queries	342
InfoSets and Structural Changes in a Logical Database	345
Logical Databases with Different Node Types	346
Interactive Multiple-Line Basic Lists.....	348
Logical Database F1S.....	349
QuickViewer	352
Functions for Managing QuickViews	353
Creating QuickViews.....	355
Selecting a Data Source	356
Executing QuickViews	357
Output Options	359
Converting QuickViews to Queries	360
Basis Mode.....	361
Selecting List Fields	362
Determining the Sort Sequence	363
Selecting Selection Fields.....	364
Example	365
Layout Mode/The Graphical Query Painter	367
Selecting List Fields	369
Selecting Selection Fields.....	370
List Display Options	371
Headers and Footers	372
List Width (Ruler)	374
List Line Output Options.....	375
Colors.....	376
Column Headers	377
Separators	378
Further Options	379
Field Display Options	380

Positioning and Sizing.....	381
Colors.....	382
Totaling and Counting.....	383
Screen Templates.....	387
Currency Fields and Quantity Fields.....	388
Further Options	389
Output Options for Control Levels.....	390
Sorting.....	391
Control Level Text.....	392
Subtotals	393
Counting Fields	394
Further Options	396
Toolbars	397

SAP Query (BC-SRV-QUE)

The InfoSet Query, the SAP Query and the QuickViewer allow you to define reports without having to program them yourself.

InfoSet Query

The InfoSet Query is a useful tool for maintaining queries within the SAP Query that is suitable for developing queries and for ad-hoc reporting.

All query information, meaning the selection criteria and output, are available on a single screen. The clear and concise arrangement of screen areas means that even the novice user can soon generate queries with confidence. Graphical interface elements and drag&drop make the tool easy to use.

InfoSet Query is suitable for reporting in all areas of the SAP R/3 system. A special feature is the **Human Resources (HR)** component. When InfoSet Query is used in HR for ad-hoc reporting, the name Ad-hoc Query is used instead of InfoSet Query. Due to the data model in HR, the InfoSet Query also contains the *Object Selection* function with which you should work in order to evaluate HR data.

SAP Query and Quick Viewer

To define a report with the SAP query, you first have to enter individual texts, such as titles, and select the fields and options which determine the report layout. You then assign a particular sequence by numbering the fields. Then you can edit list display in WYSIWYG mode whenever you want using drag and drop and the other toolbox functions available.

SAP Query is a comprehensive tool for defining reports in different forms such as basic lists, statistics, or ranked lists. In contrast, the QuickViewer is a simplified tool for generating basic lists. The QuickViewer is especially useful for beginning users and occasional use.

QuickView definitions are user-dependent. You can transfer a QuickView into SAP Query in order to make it accessible to other users, or to use the other output forms and functions available in SAP Query.

Reports created using the QuickViewer or SAP Query may also be used to pass data to external programs (Excel or MS Word, for example).

InfoSet Query

Purpose

The InfoSet Query is a useful tool for maintaining queries within the SAP Query and is suitable for developing queries and for ad-hoc reporting.

InfoSet Query is suitable for reporting in all areas of the SAP R/3 system. A special feature is the **Human Resources (HR)** component. When InfoSet Query is used in HR for ad-hoc reporting, the name **Ad-Hoc Query** is used instead of InfoSet Query. In addition, due to the data model in HR, the InfoSet Query contains the *Object Selection* function with which you should work in order to evaluate HR data. You can find further information under [HR in the InfoSet Query \[Seite 52\]](#).

All query information, meaning the selection criteria and output, is available on a single screen. The clear and concise arrangement of screen areas means that even an inexperienced user can quickly get to grips with the process of generating queries. Graphical interface elements and drag&drop make the tool easy to use.

Integration

InfoSet Query is designed in such a way that it can be called directly from roles with a relevant InfoSet and be used directly by the end-user for reporting within his/her roles. This also means that the end-user does not have to carry out any SAP Query administration functions, for example assigning him/herself to a user group.

The queries created by InfoSet Query can also be processed with other SAP Query tools. Conversely, queries that have been generated with other SAP Query tools can be processed with InfoSet Query.

Features

When you call InfoSet Query, you are first provided with a more or less detailed template (an InfoSet or an available query) that you can change or adjust however you like. You can change the selection fields and selection values used as well as the output fields and output format. These changes are temporary, meaning that all changes are lost when you leave the query (ad-hoc reporting). The advantage is that several users can use the query as a template at the same time without locking each other out. If you require further information, see [Saving Queries \[Seite 35\]](#).

It is also possible to save changes thereby overwriting the available query or creating a new query (development). The process of creating queries is always the same. An explanation of the differences between using InfoSet Query as an ad-hoc reporting tool or as a development tool can be found in the section [Calling the InfoSet Query \[Seite 46\]](#).

InfoSet Query uses the SAP List Viewer (ALV) as its standard output medium. This means that after executing a query, the output list is displayed in the ALV. There is a whole range of formatting and export options available to you, for example, sorting, filtering, export for word processing and spreadsheet programs. For more information see [SAP List Viewer \(ALV\): Grid Control \[Extern\]](#).

You can log the execution of queries in the InfoSet query if you want to. You can find more information under [Logging \[Seite 42\]](#).

Restrictions

Because InfoSet Query uses ALV as its standard output,

- every query can only output one list. Queries created by SAP Query that use a single-level list for output can contain a basic list, up to nine ranked lists, and up to nine statistics per query.

InfoSet Query

- output can only be in single-level lists. This means that you cannot distribute output fields over several lines.

You cannot define and use local fields with InfoSet Query either.

For this reason you should not process an available query with InfoSet Query if it contains the following objects:

- several lists (only the first list is displayed)
- a basic list with line groups (the first line of each is displayed)
- local fields (local fields are not displayed)

If you do use a query with one of these properties in InfoSet Query, then the parts of the output are shown as described above. These properties are lost if you overwrite the query. These properties are not lost as long as you do not save the query, and you can continue to call and execute the query with all properties using the SAP Query maintenance transaction.

See also:

[Maintaining Queries \[Seite 13\]](#)

[Calling up InfoSet Query \[Seite 46\]](#)

[HR in InfoSet Query \[Seite 52\]](#)

Maintaining Queries

Use

The overview presents the initial screen of InfoSet Query in the form of a diagram and describes the general process of generating queries.

Prerequisites

To be able to use **all** the functions of InfoSet Query, you should have started InfoSet Query from a role with the reporting type development and must be authorized to change queries.

Features

After calling InfoSet Query, you will see a screen with three areas giving you all the information you need on creating and executing a query. Apart from a few dialog boxes, this screen is the only screen of InfoSet Query.

Maintaining Queries

InfoSet Query (InfoSet BCS1, Query QUE_SCHUHM_01, Benutzergruppe MI)

Feldgruppe / Felder

Feldgruppe / Felder	Selektion	Ausgabe
Selektionsfelder aus InfoSet		
Flugverbindungen	2	3
Name einer Fluggesellschaft	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Ankunftszeit	<input type="checkbox"/>	<input type="checkbox"/>
Kurzbezeichnung der Flugge	<input type="checkbox"/>	<input type="checkbox"/>
Startort	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Zielort	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Code der Flugverbindung	<input type="checkbox"/>	<input type="checkbox"/>
Abflugzeit	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Entfernung	<input type="checkbox"/>	<input type="checkbox"/>
Masseinheit der Entfernung	<input type="checkbox"/>	<input type="checkbox"/>
Flugzeit	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Flüge	1	1
Flugbuchungen	1	

Feldname

Feldname	Option	Wert
Startort	NEW YORK	
Zielort	SAN FRANCISCO	
Flugdatum	01.11.1999	
Geschäfts-/Privatkunde	P	

Flugdatum

Flugdatum	Abflug	Airline	Flugzeit
18.07.1999	14:07:20	Airline8	0:12
29.07.1999	14:07:20	Airline11	0:12
06.08.1999	14:12:20	Airline10	0:04
	14:11:20	Airline6	0:01
08.08.1999	14:07:20	Airline5	0:08
09.08.1999	14:10:20	Airline11	0:06
27.08.1999	14:08:20	Airline1	0:12
30.08.1999	14:12:20	Airline7	0:09
01.09.1999	14:06:20	Airline7	0:03
12.09.1999	14:13:20	Airline2	0:13
10.09.1999	14:14:20	Airline15	0:07

BIE (1) (000) pwdf0033 INS

The current InfoSet is displayed in the top left-hand corner of screen. You can choose between three display options:

- Overview tree of the InfoSet with its field groups (default setting)
- Structure of the logical database
- Field catalog

You can open another screen area under the InfoSet display, to display technical information on a selected field.

The selections used as options for reading the dataset are displayed in the top right-hand corner of the screen. You have two options for displaying the selections:

- Selection control (ALV grid ready for input)
- Selection screen

The output preview is displayed in the lower area of the screen, as long as output fields have been selected. You can edit the output at this point. For example, you can change the order of the columns, form totals and subtotals or display the results list of the query.

What the sub-screens contain after InfoSet Query is called depends on the template with which you started InfoSet Query. Depending on the parameters used during call-up (see [Calling InfoSet Query \[Seite 46\]](#)) you can use a certain InfoSet or query with selection and output fields. If InfoSet Query was started with an InfoSet, you will only see the InfoSet displayed in the top left-hand corner of the screen. If InfoSet Query was started with a query, all areas of the screen are filled according to the query definition, and the defined query settings apply (for example, output as a statistic).

If no parameters were used when InfoSet Query was called, the steps so far (via the SET/GET parameters) determine which InfoSet or query was last used and InfoSet Query will be started with the relevant values. If InfoSet Query is started for the first time without parameters, a dialog box appears in which you can select the user group and an InfoSet.



Reading a query as a template does not lock the query. So, there is no restriction as to the number of users using the query as a template at the same time.

Choose *Goto* → *Variant maintenance* to maintain a query variant. The *Goto* → *Report assignment* function can be used to maintain entries in the Report-Report-Interface.



The query loaded as a template (and not saved) or the last one used is always the one to be maintained. This means that you can make the entries straight after loading a query.

See also:

[Editing the InfoSet Display \[Seite 21\]](#)

[Output List Display \[Seite 25\]](#)

[Creating New Queries \[Seite 31\]](#)

[Settings \[Seite 39\]](#)

[Calling Other Reports \[Seite 116\]](#)

Starting InfoSet Query

Use

You can start InfoSet Query from role menus, from the SAP menu or via the maintenance transaction for queries in SAP Query.

Procedure: Starting from SAP Easy Access/Role Menu

You are working with a role menu or the Standard Easy Access Menu. You have already decided in which context you want to start InfoSet Query (this is necessary because InfoSet Query can be used in the role menus as well as in the standard menu in several info systems).

1. Start *InfoSet Query* from the *Easy Access* menu with a double-click.

If you are assigned to a role that supports Reporting with InfoSet Query and you have started InfoSet Query from the role menu, InfoSet Query starts with an InfoSet and possibly a query. This means you can start directly to create or edit a query.

If you are not assigned to a role and have started InfoSet Query from a menu, you get to a dialog box first, where you have to select the query area, a user group and an InfoSet. This brings you to the InfoSet Query screen in which the InfoSet Query is displayed but no selections have been made. However, it is possible for InfoSet Query to start directly with an InfoSet or a query. If this is the case, you can start working on it straight away.



Calling InfoSet Query has been set up in HR so that InfoSet Query starts with fixed values for the parameters *Query Area* and *User Group*, and only Ad hoc Reporting can take place. This means that a user who starts InfoSet Query over HR's information systems either goes straight to the assigned InfoSet in the initial screen of InfoSet Query or can choose between several assigned InfoSets. In this context, InfoSet Query is known in this type of call-up as **Ad Hoc Query**.

Procedure: Starting from SAP Query

You are assigned to a minimum of one user group, to which, in turn, a minimum of one InfoSet is assigned.

1. In the SAP Easy Access menu, choose *Tools* → *ABAP Workbench* → *Utilities* → *SAP Query* → *Queries*.

You get to the initial screen of the maintenance transaction for maintaining queries in SAP Query.

2. To start InfoSet Query with a query, select a query.

3. Choose  InfoSet Query.

InfoSet Query is called.

If you are starting InfoSet Query for the first time since logging on the system, the dialog box for selecting InfoSets is displayed. Here, select a user group and an InfoSet.

If you have already used InfoSet Query after logging on, the user group and the InfoSet that were used last are displayed.

If you have selected a query, this is used as a template.

If you have called InfoSet Query from the standard area, you have the opportunity to use Ad hoc Reporting. If you have called InfoSet Query from the global area, you can develop queries. If you require further information, see [Type of Reporting \[Seite 48\]](#).



Please note that the selected query is not locked when you use this type of call-up for InfoSet Query. This only happens when you select a query in the maintenance transaction for queries and edit this with the *Change* function.

See also:

[Type of Reporting \[Seite 48\]](#)

[Calling up InfoSet Query \[Seite 46\]](#)

[Rights to Access InfoSet Query \[Seite 47\]](#)

InfoSet Display

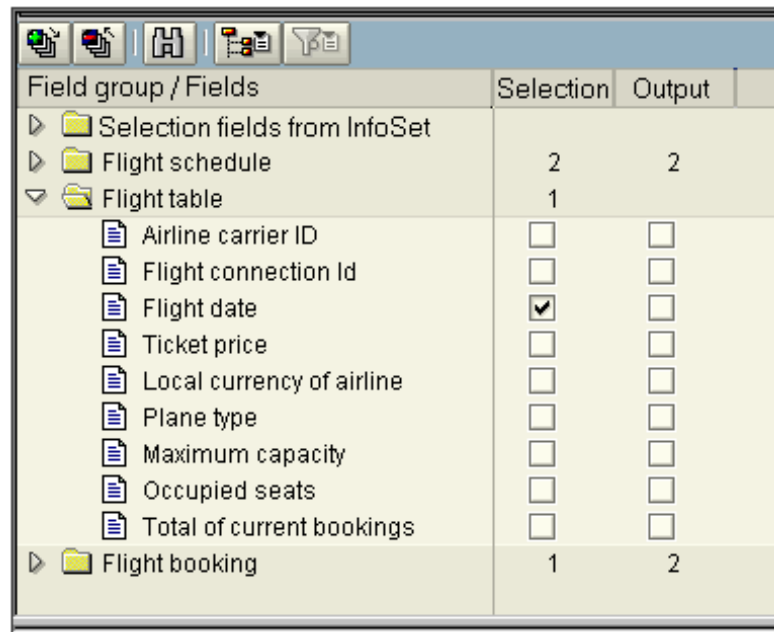
InfoSet Display

Use

The current InfoSet is displayed in the top left-hand corner of the InfoSet Query. You can choose between three display options:

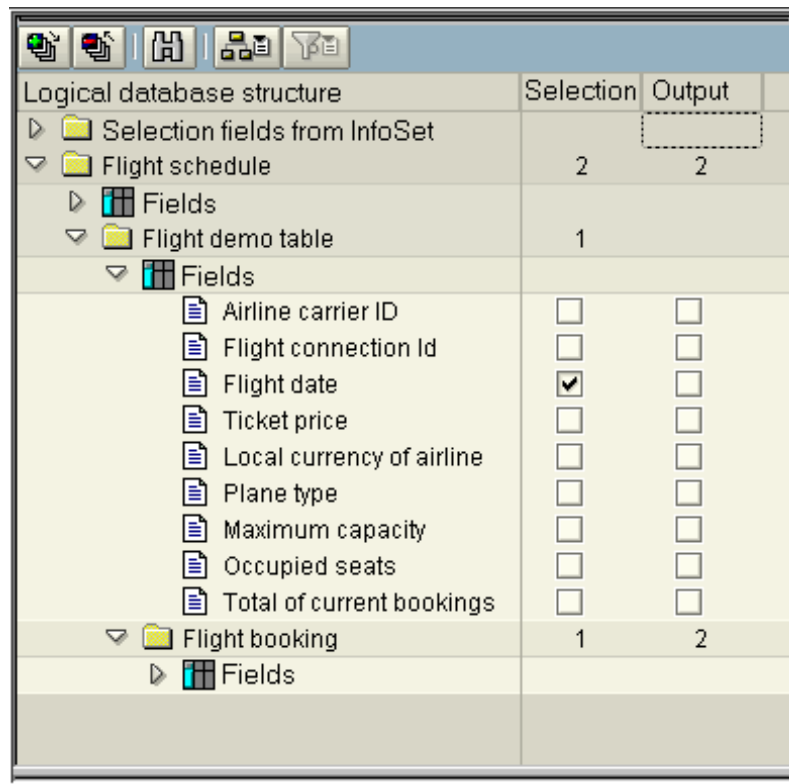
- Overview tree of the InfoSet with its field groups (default setting)

In addition to the InfoSet field groups, the additional field group *Selection fields from InfoSet* is automatically displayed. This contains the fields that were defined as selection fields in the InfoSet or in the logical database.



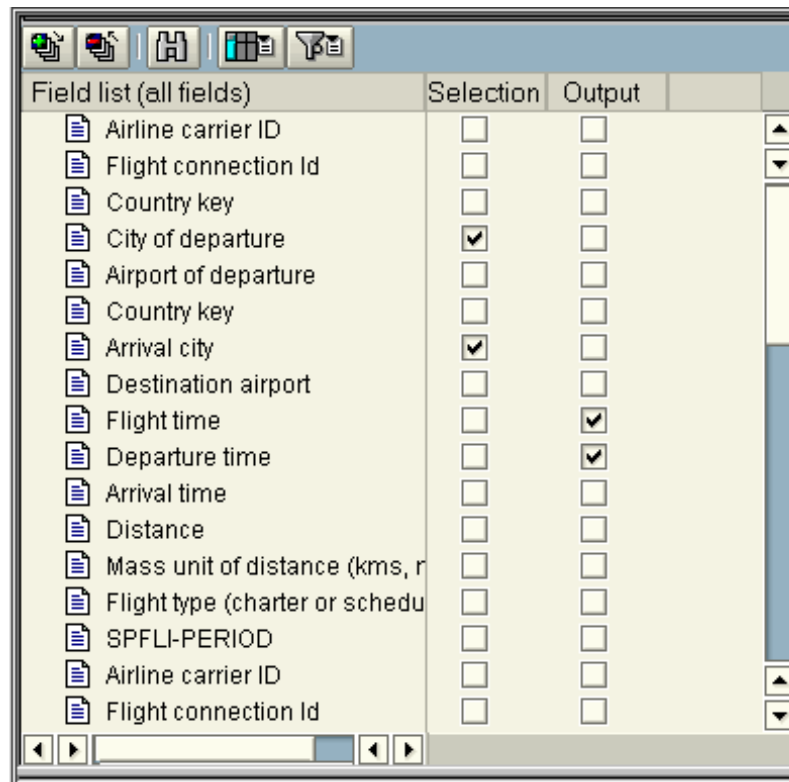
Field group / Fields	Selection	Output
▶ Selection fields from InfoSet		
▶ Flight schedule	2	2
▼ Flight table	1	
Airline carrier ID	<input type="checkbox"/>	<input type="checkbox"/>
Flight connection Id	<input type="checkbox"/>	<input type="checkbox"/>
Flight date	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Ticket price	<input type="checkbox"/>	<input type="checkbox"/>
Local currency of airline	<input type="checkbox"/>	<input type="checkbox"/>
Plane type	<input type="checkbox"/>	<input type="checkbox"/>
Maximum capacity	<input type="checkbox"/>	<input type="checkbox"/>
Occupied seats	<input type="checkbox"/>	<input type="checkbox"/>
Total of current bookings	<input type="checkbox"/>	<input type="checkbox"/>
▶ Flight booking	1	2

- Structure of the logical database



Logical database structure	Selection	Output
Selection fields from InfoSet		
Flight schedule	2	2
Fields		
Flight demo table	1	
Fields		
Airline carrier ID	<input type="checkbox"/>	<input type="checkbox"/>
Flight connection Id	<input type="checkbox"/>	<input type="checkbox"/>
Flight date	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Ticket price	<input type="checkbox"/>	<input type="checkbox"/>
Local currency of airline	<input type="checkbox"/>	<input type="checkbox"/>
Plane type	<input type="checkbox"/>	<input type="checkbox"/>
Maximum capacity	<input type="checkbox"/>	<input type="checkbox"/>
Occupied seats	<input type="checkbox"/>	<input type="checkbox"/>
Total of current bookings	<input type="checkbox"/>	<input type="checkbox"/>
Flight booking	1	2
Fields		



- Field catalog



Field list (all fields)	Selection	Output
Airline carrier ID	<input type="checkbox"/>	<input type="checkbox"/>
Flight connection Id	<input type="checkbox"/>	<input type="checkbox"/>
Country key	<input type="checkbox"/>	<input type="checkbox"/>
City of departure	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Airport of departure	<input type="checkbox"/>	<input type="checkbox"/>
Country key	<input type="checkbox"/>	<input type="checkbox"/>
Arrival city	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Destination airport	<input type="checkbox"/>	<input type="checkbox"/>
Flight time	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Departure time	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Arrival time	<input type="checkbox"/>	<input type="checkbox"/>
Distance	<input type="checkbox"/>	<input type="checkbox"/>
Mass unit of distance (kms, r	<input type="checkbox"/>	<input type="checkbox"/>
Flight type (charter or schedu	<input type="checkbox"/>	<input type="checkbox"/>
SPFLI-PERIOD	<input type="checkbox"/>	<input type="checkbox"/>
Airline carrier ID	<input type="checkbox"/>	<input type="checkbox"/>
Flight connection Id	<input type="checkbox"/>	<input type="checkbox"/>

InfoSet Display

Every InfoSet field is indicated with an icon that tells you if field values have corresponding texts. The following icons are displayed:

-  You can only use values for selection and output with these fields. You can use this field for selection as well as for output.
-  A text can be automatically determined for every possible value of the field. When selecting a field like this for selection or for output you can decide whether you want to use the field value or the text that belongs to the value or both. You can set the default by using the function [Settings \[Seite 39\]](#). If you want to make a selection that is different to the default, select the field and make the selection required in the context menu.

There are two columns for every InfoSet field in the tree, which each has one selection field. Selecting a field in the first column copies that field to the selection field list. Selecting a field in the second column turns that field into an output field.

Editing the InfoSet Display











Use

This general procedure describes the different ways of displaying InfoSets and the editing options for these display options.

Prerequisites

You are in the initial screen of the InfoSet Query. An InfoSet Query is displayed in the top left-hand corner of the screen.

Procedure

Function	Procedure/Icon	What you need to know
<i>Expand subtree</i>	Select a field group or a node of the logical database and choose  .	You cannot use this function for the display option <i>Field Catalog</i> .
<i>Compress subtree</i>	Select a field group or a node of the logical database and choose  .	You cannot use this function for the display option <i>Field Catalog</i> .
<i>Find</i>	<p>You can search for fields within the InfoSet.</p> <p>Choose .</p> <p>Enter all or part of the name of the field in the dialog box.</p> <p>Choose  in the dialog box.</p> <p>The cursor stays on the first field that contains the term you are searching; the dialog box is still on the screen. You can continue the search by choosing  in the dialog box.</p>	<p>If you are looking for parts of a field name, just enter that part in the <i>Search</i> dialog box. Do not use a masked entry, such as flight*</p> <p></p> <p>The entry flight will give you, for example, the fields <i>Flight Connections</i>, <i>Flight Details</i> or <i>Percentage of Seats Occupied Per Flight</i>.</p>
<i>Display options</i>	<p>Depending on the current settings, choose either ,  or .</p> <p>In the menu that appears select one of the display options <i>Field groups</i>, <i>Structure of logical database</i> or <i>Field catalog</i>.</p>	
<i>Display all/certain fields of a field catalog</i>	<p>Choose .</p> <p>To restrict the field catalog to fields of a certain field group, select one of the field groups in the dialog box that appears. To display all the fields, choose <i>All fields</i>.</p>	You can only use this function for the display option <i>Field Catalog</i> .

Editing the InfoSet Display

<i>Technical names on/off</i>	<p>You can display the technical names of the fields in an additional column.</p> <p>To do this, in the context menu of the InfoSet display title choose <i>Technical Names on/off</i>.</p>	
<i>Display field properties</i>	<p>You can display the properties (such as data type and length) of any field you want in an additional area of the screen.</p> <p>Select the field you want.</p> <p>Choose <i>Display field properties</i> in the context menu.</p>	

Selection Display

The query selections are displayed in the top right-hand corner of InfoSet Query. The following displays are possible:

- Editable ALV Grid Control (set by default at the start of InfoSet Query)
- Selection screen (can be used as an alternative)

You can switch between the two display options by choosing *Extras → Standard Selection Screen* or *Extras → Selection Control On*. The two display options are described in more detail below.

ALV Grid Control



	Field name	Option	Wert	
	City of departure	=	NEW YORK	
	Arrival city	=	SAN FRANCISCO	
	Flight date	>	01.11.1999	
	Business/private customer	=	P	

If you use the ALV grid to display the selections and choose a selection field, a new line appears in the selection field list. The selection field list contains six columns as follows:

- Selection column, click in this column to select the line required. You can select one or more lines. You can move selected lines to another position within the list using drag & drop, or you can delete the selected lines.
- Column 2
This column specifies whether the value or the text for value is used for the selection. You can subsequently switch between *Text*, *Value* and *Text and Value* in the text field context menu.
- *Field name*
Contains the field names.
- *Option*
Contains the selection option. You can use the pushbutton in this column to call up a selection screen for the selection options and select the option required (for example, < or >). If you do not select a selection option, = is used by default.
- *Value*
Is there for entering a selection value. If you only need one value (when using selection options = or *), you can enter that value here.
- Is there for entering multiple selections, for example, several single values or intervals.

You can also use the functions and . You can use to delete selected selection fields. You can use to check the entries you made for selection values.

Selection screen

Selection Display

Connections

Airline carrier _____ to _____

Bookings

Booking number _____ to _____

Posting date _____ to _____

☐ Display cancellations as well

Depart. city NEW YORK

Arrival city SAN FRANCISCO

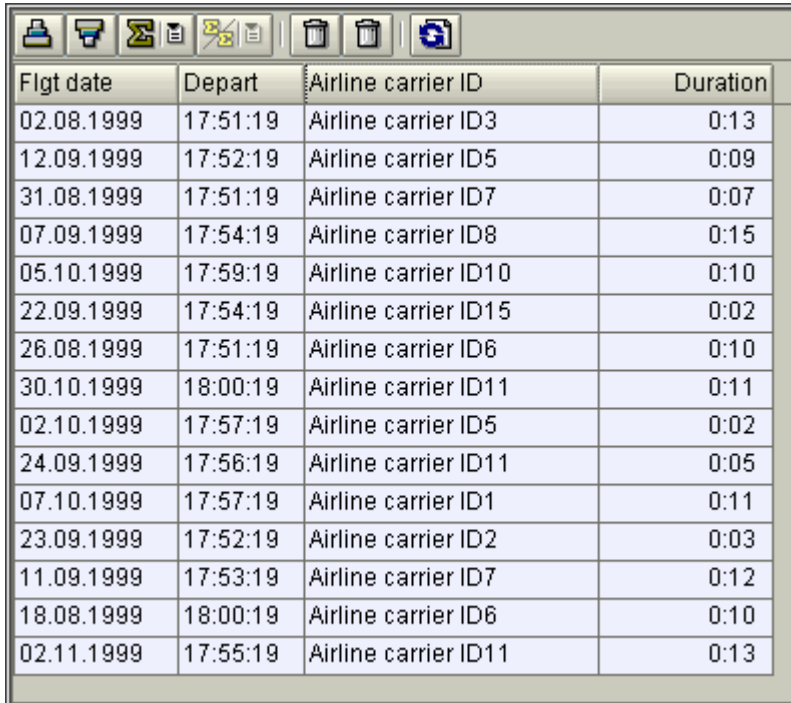
Date of flight 01.11.1999

B/P customer P

If you use the selection screen instead of the ALV Grid Control, the selection fields are displayed in the grouping determined by the logical database. Radio buttons and pushbuttons can be displayed only on a selection screen. The selection screen has the advantage that a check is run as soon as the values are entered.

Output List Display

The output list is displayed in the lower area of the InfoSet Query screen in the SAP List Viewer or in full screen. If you have selected an output field, you will first see an output preview that contains example data.



Flgt date	Depart	Airline carrier ID	Duration
02.08.1999	17:51:19	Airline carrier ID3	0:13
12.09.1999	17:52:19	Airline carrier ID5	0:09
31.08.1999	17:51:19	Airline carrier ID7	0:07
07.09.1999	17:54:19	Airline carrier ID8	0:15
05.10.1999	17:59:19	Airline carrier ID10	0:10
22.09.1999	17:54:19	Airline carrier ID15	0:02
26.08.1999	17:51:19	Airline carrier ID6	0:10
30.10.1999	18:00:19	Airline carrier ID11	0:11
02.10.1999	17:57:19	Airline carrier ID5	0:02
24.09.1999	17:56:19	Airline carrier ID11	0:05
07.10.1999	17:57:19	Airline carrier ID1	0:11
23.09.1999	17:52:19	Airline carrier ID2	0:03
11.09.1999	17:53:19	Airline carrier ID7	0:12
18.08.1999	18:00:19	Airline carrier ID6	0:10
02.11.1999	17:55:19	Airline carrier ID11	0:13

You can format the output preview. This enables you to determine how data is arranged when output (for example, the sequence of columns and output list sorting), and whether data is also aggregated (for example, summation of numerical columns).

Furthermore, you can select a *Type of Output List* by choosing *Edit* → *Settings*. This determines if and how data is aggregated when output. You can select one of the following options:


- Basic list
- Statistics
- Ranked list

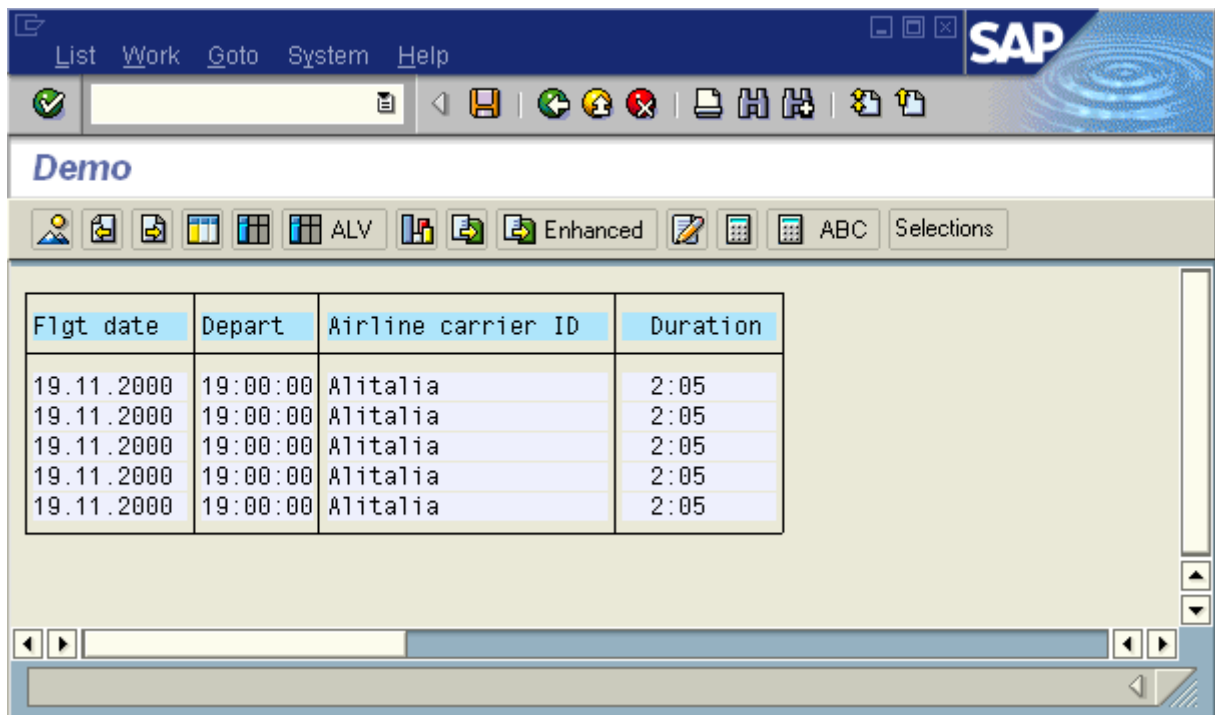
The settings and formatting are visualized in the output preview, and apply to the data output in the InfoSet Query screen, as well as to the data output in full screen.

You can display the real data in this area of the screen by choosing the *Refresh* function. You can then use the SAP List Viewer functions. For more information see [SAP List Viewer \(ALV\): Grid Control \[Extern\]](#).

Output List Display

Flgt date	Depart	Airline carrier ID	Duration
19.11.2000	19:00:00	Alitalia	2:05
20.11.2000	19:00:00	Alitalia	2:05
21.11.2000	19:00:00	Alitalia	2:05
22.11.2000	19:00:00	Alitalia	2:05
23.11.2000	19:00:00	Alitalia	2:05








You can display the output list in full screen by using the  *Output* function. In full screen you can use other output formats in addition to those of the SAP List Viewer, for example, the default list.





Flgt date	Depart	Airline carrier ID	Duration
19.11.2000	19:00:00	Alitalia	2:05
19.11.2000	19:00:00	Alitalia	2:05
19.11.2000	19:00:00	Alitalia	2:05
19.11.2000	19:00:00	Alitalia	2:05
19.11.2000	19:00:00	Alitalia	2:05

Formatting the Output

You can format the output preview in the InfoSet Query screen. When you execute the query, the formatting applies to the output list display in the InfoSet Query screen and to the output in the full screen. You can use the following options:

Function	Procedure/Icon	What you need to know
<i>Sort ascending</i> <i>Sort descending</i>	<p>Select a column, and sort ascending by choosing  or descending by choosing .</p> <p>You can use several sort criteria.</p> <p>To do so, do not select a column, and then choose  or . You reach the <i>Define Sort Order</i> dialog box. You can select any output fields as sort criterion, and decide whether to sort according to this criterion in ascending or descending order.</p>	
<i>Total</i>	<p>Select the column you want, and choose .</p> <p>A total is output at the end of the column.</p>	<p>You can create totals only for columns that contain numerical values.</p> <p>If you also want to output subtotals, you must first create totals using the numerical column.</p>
<i>Subtotals</i>	<p>Select the column that contains the criterion for subtotals, and choose .</p> <p>If the criterion for subtotals changes, a subtotal is output.</p>	<p>You can use columns as a criterion for subtotals only if they contain non-numerical values. The list is automatically sorted according to this criterion.</p> <p>If you report on the annual salaries of your employees, for example, you can create totals for the <i>Annual Salary</i> column and output a subtotal for each organizational unit.</p>
<i>Delete field(s)</i>	<p>Select the column you want to delete from the output list.</p> <p>Choose .</p>	<p>If you do not select a column, this function deletes all output fields.</p>

Formatting the Output

<i>Initialize formatting</i>	Choose  .	Resets output formatting to its original state. Use this function if you no longer need to create subtotals or totals. The column sequence is not reset.
<i>Update data</i>	Choose  . The query is executed and the results list is displayed in the lower part of the InfoSet Query screen.	After the query has been executed, you can use the ALV functions from the output on the InfoSet Query screen, for example, Excel Inplace display or the export for text processing programs. If you change the output list in such a way that a new selection must take place (when you add a new field, for example,) you see the example dates in the output list again. You need to update the data again so that the query is executed again.
<i>Change column sequence</i>	Select the column whose position you want to change, and use drag & drop to move it to the required position.	
<i>Output / do not output currency or unit</i>	In the basic list, currencies or units are output automatically in an additional column after the value column. Choose the following in the context menu of the value column, to change the position of this column: <i>Currency → Before the value or After the value</i> <i>Unit → Before the value or After the value</i> Choose the following in the context menu of the value column, to prevent the currency or unit column from being output: <i>Currency → Do not output</i> <i>Unit → Do not output</i>	An additional column for currencies or units is not output in statistics or ranked lists. Choose <i>Edit → Settings</i> on the tabstrip <i>Statistics/Ranked List</i> if you want to enter a reference currency to which all currency fields are converted.

Formatting the Output

<i>Enhance statistics</i>	<p>You can enhance statistics with the following columns:</p> <ul style="list-style-type: none"> • Mean values • Percentage-based distributions • Number of processed data records <p>Choose <i>Output mean value</i>, <i>Output share in %</i>, or <i>Output total number</i> in the context menu of the value column.</p>	These enhancements are only possible for numerical columns.
---------------------------	--	---

The various output lists include further options to enable you to edit output. For more information on how to edit output lists, see [Interactive List Display Functions \[Seite 109\]](#) and [Interactive Functions for Further List Processing \[Seite 117\]](#). For more information on how to output data in the SAP List Viewer, see [SAP List Viewer \(ALV\): Grid Control \[Extern\]](#).

Reusing Queries

Use

You can save queries that you need frequently. You can then open, and execute them again in InfoSet Query.

- If you frequently use a query with the same selection fields but different values, you should save it **without** values.
- If you frequently use a query with the same selection fields and values, you should save it with a variant, that is, **with** values and selection options.

You can also use saved queries as templates for new queries. To do this, start InfoSet Query with a query, or open an existing query, edit it, and save it with a new name.

You can find more information on the functions *Save* and *Save as* in InfoSet Query under [Saving Queries \[Seite 35\]](#).

Integration

SAP Query enables you to access and process queries defined in InfoSet Query.

Creating New Queries

Use

You can use InfoSet Query for flexible data analysis that you cannot carry out with standard reports. By simply choosing selection and output fields, you can define the report you need, and display a results list. You can repeat this process as often as you like. You do not require programming skills to create reports using InfoSet Query.

You can also save queries, make queries you have saved available for other users to use, and transport queries, providing you have the relevant authorization.

Prerequisites

You are in the initial screen of InfoSet Query. Depending on how you have started InfoSet Query,

- an InfoSet is available
- a query with selections and output fields is available
- you first have to select a user group and an InfoSet from a popup

Procedure

1. In the column *Selection*, select the relevant selection fields and, if necessary, remove the selection fields you do not need by clicking the relevant check box.

The system transfers the selected fields to the selection field list. The selection fields that you have removed are no longer in the list.

2. In the column *Output*, select the relevant output fields and, if necessary, remove the output fields you do not need by clicking the relevant check box.

The system transfers the selected fields to the output preview. The output fields that you have removed are no longer in the output preview.





You can also add the fields to the relevant area (selection or output area) using drag & drop. If you use drag and drop, you can insert the fields directly into the correct position in the output preview.

Depending on the default settings, fields with texts have only their text or their value transferred when you select them by ticking the check boxes or by using drag & drop. However, you can also select fields by using the context menu of the relevant field. Here, you can choose whether you want to use the value, the text, or both.



3. Enter the relevant selection values for the selection fields in the column *Value*.

The value types (numeric, alphanumeric, date) that you can enter for the selection fields depend on the type of selection field.

4. In the *Options* column, choose selection options (such as < or =).

5. To execute the query, choose  or  *Output*.

Result

The query is executed. If you started it by choosing , the results list is displayed in the lower part of the screen. If you started it by choosing  *Output*, the results list is displayed in full screen.

Creating New Queries

The default output works as follows; no selection screen is displayed and the specified selections are taken into account when outputting. You can, however, change these standard settings. You can find more information under [Settings \[Seite 39\]](#).


You can save the query, if you are authorized to do so. You can find more information under [Saving Queries \[Seite 35\]](#), [Type of Reporting \[Seite 48\]](#) and [Rights to Access InfoSet Query \[Seite 47\]](#).

Loading Queries


Use

InfoSet Query enables you to access saved queries.


Procedure

1. Choose .

This takes you to the *Open Query* dialog box. It contains a list of all queries and variants that are assigned to the current user group.

2. If you want to load a query that was created under a different user group, change the user group.
3. Select the required query.
4. Choose .

Result

The query is opened. You can now execute the query. You can also change and save the query as required. If you want to overwrite the original version, choose *Query* → *Save* or . If you want to assign a new name to the changed version, choose *Query* → *Save as*.



The last five queries that you saved during an R/3 session are listed in the *Query* menu and you can load them from there.

Deleting Queries

Deleting Queries

Use

InfoSet Query enables you to delete queries that are no longer required.



The query that you delete is always the query you loaded as a template.

Prerequisite

You have loaded a query. You can check this in the header of the InfoSet Query. If you have loaded a query, the name of the query is displayed here.

Procedure

1. Choose *Query* → *Delete*.
2. To confirm, choose *Yes*.

Result

The query that you loaded into the InfoSet Query is deleted. However, the current InfoSet, selections and output fields are still displayed after the query is deleted.

Saving Queries

After starting InfoSet Query you can change a template as often as you like and execute the corresponding query. If and where you can save a query depends on your authorizations and on the reporting you are authorized to carry out. If you require further information, see [Rights to Access InfoSet Query \[Seite 47\]](#) and [Type of Reporting \[Seite 48\]](#).

If you are authorized to save queries (authorization object `S_Query` with the value `02` for the field `Actvt`) you can use the functions `Save` and `Save as`. The way these functions work depends on your previous work with InfoSet Query.

- **Save (first usage)**

This function will take you to a dialog box where you can enter a new query name and a description. The system proposes a new name if the template you called up was an InfoSet; if it was a query, then the system proposes the query name. If you save a query under the same name again (meaning, overwrite it) then it is locked for other users. That is, other users of InfoSet Query can continue to use the locked query as a template, but cannot overwrite it.

If you are working in a context, which uses the SAP Query user groups, you can also select, from the dialog box, a user group to which you want to assign the query. At this point, you can switch between user groups to which you are assigned **and** which have access to the current InfoSet.

- **Save (if not the first time)**

If you are using the function again with a changed query during a session with InfoSet Query, the query you saved before is overwritten. The dialog box is not shown again. Locks that already exist remain valid.

- **Save as**

This function primarily exists for you to copy queries and edit them afterwards. With this function the dialog box is always shown to enter the query name and a description. The system proposes the name of the query that you used as template. The query is saved and locked under the name entered. If you have already saved and therefore locked the template, this lock is removed.

If you are working in a context, which uses SAP Query user groups, you can also select, from the dialog box, a user group to which you want to assign the query. At this point, you can switch between user groups to which you are assigned **and** which have access to the current InfoSet.



If you select an existing query from the list, it is overwritten when you confirm.

The query is locked when you save it. This means that other users, who may be working on the query at the same time, can continue to use the old version as a template in the current session with InfoSet Query, but they can no longer save the query under its original name. This also means that you can only save a query under a certain name if this query is not locked by another user.

If you have locked a query by saving it and then select a new template for InfoSet Query (InfoSet or query) the lock on the query used before is removed and another user can save the query in a changed version.

If you are not authorized to change queries, you can start the InfoSet with templates, change and execute queries, but you cannot save the changes.

See also:

Saving Queries

[Saving a Query \[Seite 37\]](#)

Saving a Query


Use

You can use InfoSet Query to save queries that you want to use again.

Prerequisites

You have defined a new query. You are authorized to save queries.

Procedure

1. Choose .
2. If you are saving the query for the first time, the dialog box *Save Query* is displayed. A new name is proposed for the query if you started the InfoSet Query with an InfoSet. The name of the query is proposed if you started the InfoSet Query with a query. You can change the names as required.
3. If necessary, change the name or title of the query.
4. If necessary, choose the *User Group* to which the new query is to be assigned. At this point, you can switch between user groups to which you are assigned **and** which have access to the current InfoSet.

Result

The query is saved and locked for other users.



You can save a changed query under the same name by choosing *Query* → *Save*, or under a new name by choosing *Query* → *Save as*. If you require further information, see [Saving Queries \[Seite 35\]](#).

Change InfoSet

Change InfoSet



Use

While working with InfoSet Query, you can access a different InfoSet as a template for queries provided that you have the appropriate authorizations.

Prerequisites

You are on the InfoSet Query screen. You have access to more than one InfoSet.

Procedure

1. Choose .
This takes you (possibly via a confirmation prompt) to the *Create New Query - InfoSet Selection* dialog box. The *User Groups* selection list contains all of the user groups to which you are assigned.
2. Select the user group that has access to the required InfoSet.
The *InfoSets* area lists all of the InfoSets that can be accessed by the selected user group.
3. Select the InfoSet that you want to use for the query.
4. Choose .

Result

This takes you back to the InfoSet Query screen. You can use the InfoSet to create queries.

Settings

Use

With the settings, you can determine:

- whether the data is displayed as a list (basic list) or aggregated before output (statistics, ranked lists) and which output form is used,
- whether the query is started using a selection screen,
- which reference currency and which reference unit is used for statistics and ranked lists,
- how many data records are displayed in a ranked list,
- how data is exported to Crystal Reports,
- which default settings should apply for using fields with texts.

Prerequisites

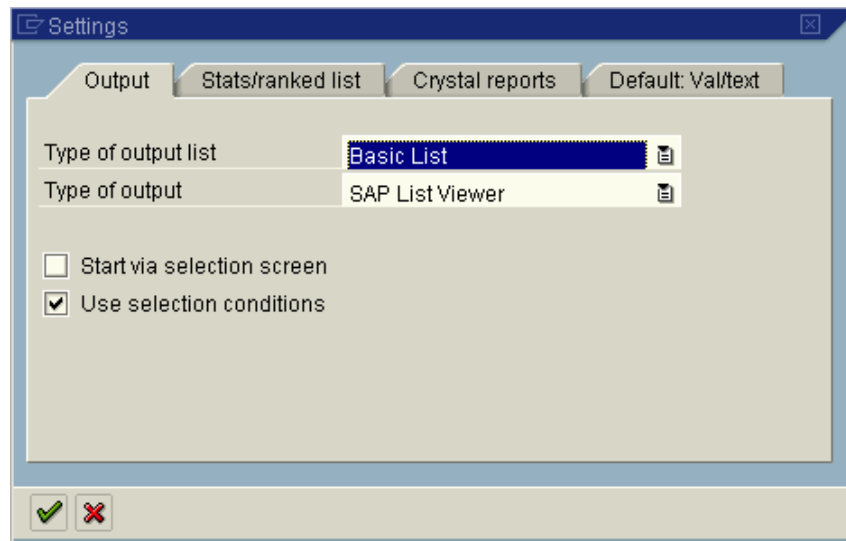
If you want to export the data to Crystal Reports, you must have them installed on your local PC.

Activities

1. On the InfoSet Query screen, choose *Edit* → *Settings*.

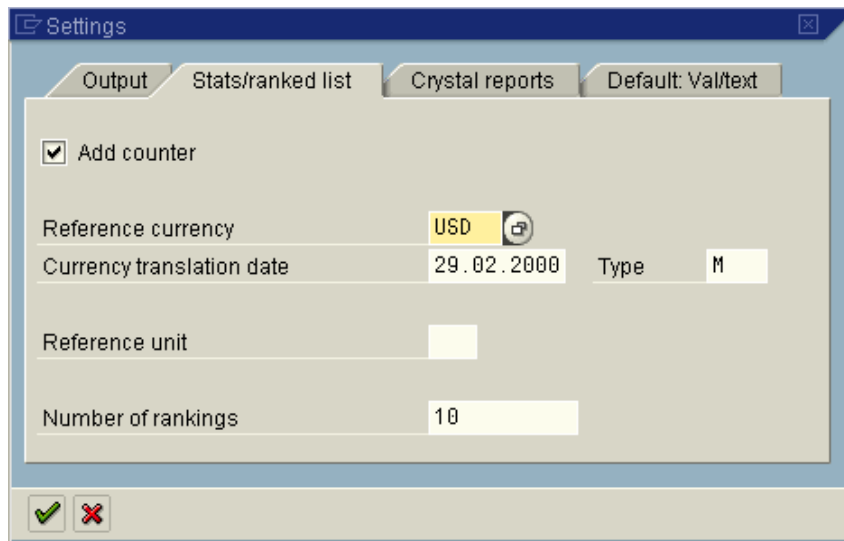
You get to the dialog box *Settings*.

On the *Output* tabstrip, you can determine the type of output list and the output format as well as whether the query is to be started via a selection screen.



On the *Statistics/Ranked List* tabstrip you can determine the reference currency for the conversion of currency fields as well as the reference unit for converting units of measurement. Also in this tabstrip, you can determine how many ranks are displayed in a ranked list.

Settings

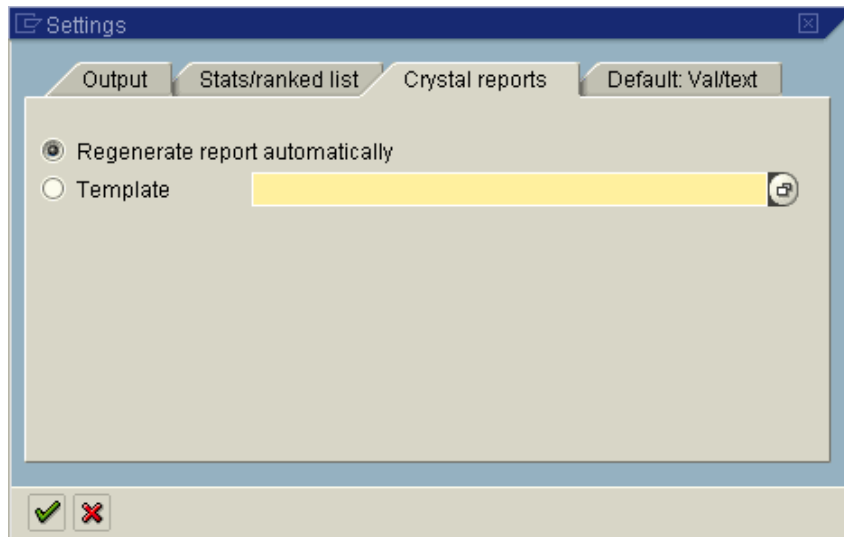


The screenshot shows the 'Settings' dialog box with the 'Crystal reports' tab selected. The 'Output' tab is also visible. The 'Add counter' checkbox is checked. The 'Reference currency' is set to 'USD'. The 'Currency translation date' is '29.02.2000'. The 'Type' is 'M'. The 'Reference unit' is empty. The 'Number of rankings' is '10'. There are 'OK' and 'Cancel' buttons at the bottom.

The tabstrip *Crystal Reports* allows you to determine how the Crystal Reports Export runs if Crystal Reports start directly from InfoSet Query. You can create a new Crystal Report every time you export, or export the data to an existing Crystal Report on the local PC.

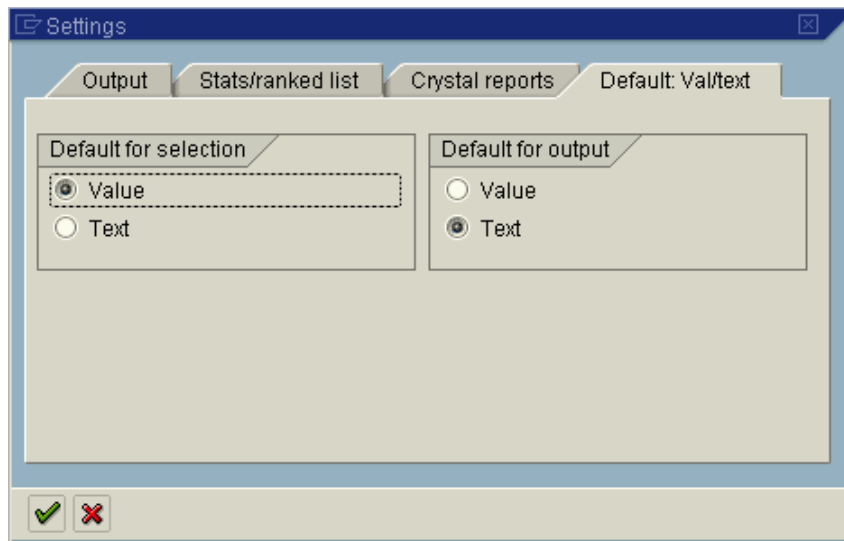



If you export the results list of the query over the output list (ALV) to Crystal Reports, these settings are overridden and those settings that were made in the ALV apply.



The screenshot shows the 'Settings' dialog box with the 'Default: Val/text' tab selected. The 'Regenerate report automatically' radio button is selected. The 'Template' radio button is also visible. There are 'OK' and 'Cancel' buttons at the bottom.

The tabstrip *Default: Value/Text* allows you to determine whether the value or the text is used when you select a field using either drag & drop or the check boxes. You can override these proposals by choosing the context menu when you select a field.



2. Enter the required options.
3. Choose .

Result

The entries are transferred for the current query.

When you have set up the Crystal Reports Export, the path for the *Crystal Report* template is saved with the query. Whereas the template itself is not saved with the query.



It you want the query **and** the Crystal Reports template to be available for other users, save the template in a directory that can also be accessed by these users. Otherwise, other users are not able to use the query and the template because the template is not saved with the query.

Logging

Logging

Use

You can log the execution of queries with InfoSet Query. This enables you to determine which InfoSet Query users have analyzed system data when, and with which selection criteria.

Prerequisites

The log is activated and the query is started in InfoSet Query. This means you use one of the following functions in InfoSet Query:

- *Refresh Data* (in the output preview)
- *Output*
- *Select* (only with HR queries)

The log is not used in other cases, that is, if you start the query as an executable program over a menu entry or via the maintenance transaction for queries in SAP Query.

Features

The following information is logged when you execute an InfoSet:

- the query area
- the InfoSet used
- the user
- the date
- the time
- all selection fields including the selection values and selection options
- all output fields

The output is not included in the log.

With the help of the Badi AQ_QUERY_PROT, you can attach your own log, where you can create a filter or another folder for the log.

You can find further information on logging and on the Badi in the Basis IMG under *Basis Components* → *SAP Query* → *Logging*.

Activities

You can activate the log by choosing the relevant function in Customizing of the SAP Query or via the maintenance transaction for InfoSets of the SAP Query (see [Activating Logging \[Seite 43\]](#)).

When you execute an InfoSet Query, the specified data from the system is logged.

You can analyze the logs using a Query or by using the Data Browser (see [Analyzing Logs \[Seite 44\]](#)).

You can delete log data no longer required by choosing the relevant function in Customizing of the SAP Query or via the maintenance transaction for InfoSets of the SAP Query (see [Deleting Logs \[Seite 45\]](#)).

Activating Logging

Use

Logging serves to log the execution of queries with InfoSet Query. You can determine for which query areas of the query and for which InfoSets a log should be carried out.

Prerequisites

You are in SAP Query, in the initial screen of the maintenance transaction for InfoSets.

Procedure

1. Choose *Extras* → *Set Log*.

You get to the screen *Change Query Log Control: Overview*. You can maintain entries for the database table AQPROTCUST in this screen.

2. Choose *New entries*.

3. Enter the data required.

If you want to log query execution within a certain query area for certain InfoSets only, enter a record for every InfoSet.

If you want to log query execution within a certain query area for all InfoSets, enter the identification for the work area and * for the InfoSet.

You cannot use the entry * for the query area. If you want to log the use of all InfoSets on execution of queries with the InfoSet Query in the standard as well as in the global area, for example, enter each query area in a separate line. For the InfoSets enter * respectively.

4. Choose .

Result

The logging is active according to what you have entered. If you start queries within InfoSet Query, their execution is logged from now on. You can analyze the logs with specific queries or by using the *Data Browser*. You should regularly delete logs that you no longer need.

See also:

[Analyzing Logs \[Seite 44\]](#)

[Deleting Logs \[Seite 45\]](#)

Analyzing Log Data



Use

You can analyze the InfoSet Query log data with specific queries or by using the data browser of the ABAP Workbench. In the global area, SAP delivers the user group /SAPQUERY/SQ and the InfoSet /SAPQUERY_LOGGING as well as some queries with which you can analyze log data. You can, of course, create queries of your own for analyzing the log data if you wish.

Prerequisites



The system contains logs on the execution of queries with InfoSet Query.

Procedure: Log analysis with a query

1. In the SAP Easy Access menu, choose *Tools* → *Utilities* → *SAP Query* → *Queries*.
You reach the screen *User Group Queries <User Group>: Initial Screen*.
2. Make sure that you are working in the global area with the user group /SAPQUERY/SQ and the InfoSet /SAPQUERY_LOGGING.
3. Select the required query.
4. Choose .
5. Enter the data required for selecting the log data.
6. Choose .

The log data corresponding to your criteria is displayed.

Procedure: Log analysis using the data browser

1. In the SAP Easy Access menu, choose *Tools* → *ABAP Workbench* → *Overview* → *Data Browser*.
You reach the screen *Data Browser: Initial Screen*.
2. Enter the table name **AQPROT**.
3. Choose .
4. Enter the data required for analyzing the log.
5. Choose .

The log data corresponding to your criteria is displayed.

Deleting Log Data


Use

You can delete log data that is no longer required for working with InfoSet Query. We recommend that you regularly delete log data that you no longer require. This ensures that space is not wasted on the database.

Prerequisites

You are in InfoSet Query, in the initial screen of the maintenance transaction for InfoSets.

Procedure

1. Choose *Extras* → *Manage log*.
This takes you to the *Deleting Logged Query Activities* screen. This screen lets you determine which logs you want to delete. You can determine, for example, whether you want just the log data to be deleted that is created for a certain InfoSet or a certain user.
2. Enter the data required.
3. Choose .

Result

The log data that corresponds to the criteria entered is deleted from the database.

Calling the InfoSet Query

Purpose

The Administrator can make the InfoSet Query available to users in a number of ways.

- For example, the call can be implemented using four reports that determine the particular call parameters. These reports decide which access rights the user has, for example, from which role and with which InfoSet the InfoSet Query can be started and which reporting type can be used (development or ad-hoc reporting).
- The InfoSet Query can also be started from the maintenance transaction for queries in the SAP Query.

See also:

[Type of Reporting \[Seite 48\]](#)

[Rights to Access InfoSet Query \[Seite 47\]](#)

[Modules for Calling the InfoSet Query \[Seite 49\]](#)

[Starting InfoSet Query \[Seite 16\]](#)

Rights to Access InfoSet Query

The Administrator can control access rights to the InfoSet Query using roles or SAP Query user groups.

- **Roles**

The technical background to controlling access authorization using roles is as follows:

- Exactly one SAP Query user group must be assigned to a role.
- InfoSets have to be assigned to this user group. Users from the user group, and now also the role, can use these InfoSets for reporting.
- It is not necessary to enter the users into the user group as they are assigned in the role assignment. This means that a user who is assigned to a role is automatically copied into the user group assigned to the role. This assignment is only valid, however, if the user uses the role to call the InfoSet Query. If InfoSet Query is accessed in any other way, then the general access authorization implemented using SAP Query user groups applies.
- To be able to save queries, a user needs the relevant 'change' authorization for queries (authorization object `S_QUERY`, for the field `ACTVT` value 02). If the user does not have this authorization, he or she is able to use the available InfoSets and queries within the role, but is not able to save queries.

You can find further information under [Including InfoSet Query in Roles \[Seite 51\]](#).

- **User groups**

The procedure for controlling access rights using user groups is exactly the same as with the SAP Query. This means that, per user group, you determine which InfoSets can be accessed and which users belong to a user group. Users allowed to create and change queries, must be given the 'change' authorization (authorization object `S_QUERY` for field `ACTVT` value 02). You can find further information under [SAP Query Authorizations \[Seite 81\]](#).

Type of Reporting

Type of Reporting

The following types of reporting are available in the InfoSet Query:

- Development

The reporting type *Development* is used for creating queries that are going to be used on a long-term basis. This means that queries can be saved (as executable programs), incorporated into menus, or distributed. Technically speaking, this means that the global area is always used, and a connection to the Workbench Organizer exists whenever you call up the InfoSet Query for the *Development* reporting type.

- Ad-hoc Reporting

The reporting type *Ad-hoc reporting* is used primarily as a quick and simple way of creating specific reports that, as a rule, are not needed more than once. Ad-hoc reporting is possible in both query areas (global and standard), and you can also save your queries. In the global area, they are always saved in a temporary development class, and there is no dialog with the Workbench Organizer. In the standard area, queries can only be transported with the tools intended for the task. You can find further information on creating and transporting queries in the SAP Query documentation under the section [Transporting Query Objects \[Seite 303\]](#) and its subsections.



Calling up the InfoSet Query has been set up in Human Resources (HR), so that the InfoSet Query has default values in the specified *query area* and *user group* parameters, and only ad-hoc reporting can take place. This means, that a user, who starts the InfoSet Query using HR's information systems, either goes straight to the initial screen of the InfoSet Query, where the assigned InfoSet is displayed, or can choose between several assigned InfoSets. In this type of call in HR, the InfoSet Query is called **Ad-hoc Query**.

The type of reporting depends on the report used to start the InfoSet Query. You can find further information on the reports for starting the InfoSet Query under [Modules for Calling the InfoSet Query \[Seite 49\]](#).

Modules for Calling the InfoSet Query

To call the InfoSet Query, the SAP System provides you with four reports and four function modules of the same name, which implement the InfoSet Query call and determine access rights and reporting type for the InfoSet Query.

The following reports/function modules are available:

- **SAP_QUERY_DEVELOPMENT_ROLE: Access using role, development**
 Parameter
 - Role (input necessary)
 - InfoSet (optional)
 - Query (optional)

A user group from the global area has to be assigned to the role. If you specify an InfoSet, then it has to be assigned to the user group. If you specify a query, then it has to be assigned to the user group.
- **SAP_QUERY_AD_HOC_ROLE: Access using role, ad-hoc reporting**
 Parameter
 - Role (input necessary)
 - InfoSet (optional)
 - Query (optional)

A user group from either the global or standard area has to be assigned to the role. If you specify an InfoSet, then it has to be assigned to the user group. If you specify a query, then it has to be assigned to the user group.
- **SAP_QUERY_DEVELOPMENT: Access using user group, development**
 Parameters (all optional)
 - User group
 - InfoSet
 - Query

The user group has to come from the global area. If you specify more than one parameter, then they must match, meaning that the InfoSet has to be assigned to the user group and the query must be based on the specified InfoSet. Input help can support you with the entries.
- **SAP_QUERY_AD_HOC: Access using user group, ad-hoc reporting**
 Parameters (all optional)
 - Query area (standard setting: Standard area)
 - User group
 - InfoSet
 - Query

First, decide which query area you want to work in with the InfoSet Query (standard setting: Standard area). To enable you to work in the global area, select the *Global Area* indicator. Enter the remaining parameters.

If you specify more than one parameter, then they must match, meaning that the InfoSet has to be assigned to the user group and the query must be based on the specified InfoSet. Input help can support you with the entries.

Modules for Calling the InfoSet Query

Calling up the InfoSet Query has been set up in Human Resources (HR), so that the InfoSet Query has default values in the specified *query area* and *user group* parameters, and only ad-hoc reporting can take place. This means, that a user, who starts InfoSet Query using HR's information systems, either goes straight to the initial screen of the InfoSet Query, where the assigned InfoSet is displayed, or can choose between several assigned InfoSets. In this type of call in HR, the InfoSet Query is called **Ad-hoc Query**.

The optional parameters are used to prepare an InfoSet or a query when calling the InfoSet Query. These "templates" can be processed directly. If no parameters are used for the call, then the system tries to come up with a suitable proposal for starting the InfoSet Query from those objects (role, user group, InfoSet) last used by the user.

The reports call the function modules with the same name directly. When calling the InfoSet Query, it does not matter if the reports or the function modules are used. To implement the call using menu entries, however, you have to use the reports.

Including InfoSet Query in Roles

Purpose

By including the InfoSet Query in a role, you ensure that the InfoSet Query is started with an InfoSet (and a query), and can therefore be easily used for appropriate reporting within the role.

Process Flow


1. Select a user group that is to be connected to the role, or create a new user group. Assign all InfoSets that are to be made available to the role user for reporting, to this user group.



If you want the role to support the development of queries (no ad-hoc reporting), then you have to use a user group assigned to a non-temporary development class from the global area. The InfoSets assigned to the user group must also have non-temporary development classes. This way you ensure that it is possible to develop queries that can be transported.



You can change the assignment of InfoSets to user groups at any time. It is not necessary to assign users.

2. Depending on the reporting type you want to use, create a variant for the report `SAP_QUERY_DEVELOPMENT_ROLE` or `SAP_QUERY_AD_HOC_ROLE`, by entering the name of the relevant role in the field *Role*. You can use the other parameters as and when you need them.
3. Assign the relevant user group to the role. To do this, call role maintenance (transaction PFCG) and select *Change*. Select the key `SAP_QUERY_USER_GROUP` by double clicking on the tabstrip *Personalization*. You get to a dialog box where you enter the user group.
4. Include the report `SAP_QUERY_DEVELOPMENT_ROLE` or `SAP_QUERY_AD_HOC_ROLE` along with the variant you created previously in the role menu. To do this, call role maintenance (transaction PFCG), select the relevant role, and choose *Change*. Select  *Report* from the *Menu* tabstrip. Insert the relevant report into the role menu (as an ABAP report) with the variant you created earlier. You should always set the indicator *Skip selection screen*.

Result

The InfoSet Query is available as a menu entry for the role. The available InfoSets are determined using the user group assigned to the role. Using the other report parameters, you determined the specifications used to start the query.

HR in InfoSet Query

Purpose

You can use InfoSet Query to report on data from Human Resources (HR) by using InfoSets based on HR logical databases (PNP, PCH, PAP).

InfoSet Query is used in HR to create reports to meet requirements that are not satisfied by standard reports. By selecting selection fields and output fields, you can access data stored anywhere within the Human Resources System. You do not require programming skills to create reports using InfoSet Query.

InfoSet Query supports two types of reporting:

- For ad-hoc reporting
- For query development

InfoSet Query has been integrated with HR information systems to enable you to perform ad-hoc reporting; that is, you can create and save queries in the standard query area. However, there is no transport link to the Workbench Organizer. In this context, InfoSet Query is known in HR as

Ad Hoc Query.

There are two ways of using InfoSet Query in HR:

- With object selection

Before data is output, a set of objects is selected for which you can output data as required. You can edit the selected set of objects before output. For example, you can display the list of objects, display details on individual objects, use the set of objects as a new reporting set, or relate two sets to each other.

This method of working has the advantage of good system performance because objects are selected using a selection routine that is particularly suitable for Human Resources.
- In basic mode (object selection is switched off)

In basic mode, the report's list of results is output immediately without a hit list being generated first.

This method of working has the advantage of enabling you to use all InfoSet fields for selection purposes. Working with object selection restricts these options.

The following sections describe how to work with object selection. Always use object selection to create reports for HR. If you require further information on how to work with InfoSet Query in basic mode, see the documentation on [maintaining queries \[Seite 13\]](#).

Integration

- You can use InfoSet Query to continue processing and report on sets of persons selected using Manager's Desktop or HIS. You can start InfoSet Query directly from these applications. The selected set of persons is transferred to InfoSet Query as a reporting set.
- You can branch from InfoSet Query to general reporting. In this case, you only use the first InfoSet Query level, that is, you select an object set. To report on this hit list, you use a standard report.
- You can enhance queries created using InfoSet Query by adding features that are only supported by SAP Query. If you require further information, see [InfoSet Query \[Seite 11\]](#).

Features

If object selection is switched on, working with InfoSet Query consists of two steps:

- In the first step, you select a set of objects (= hit list) in accordance with selection criteria that you can specify as required. You can then process the hit list. You can also use InfoSet Query to create two hit lists. You can add them together, subtract one from the other, or use them to create intersections.
- In the second step, you output data as required for the objects selected in the first step. You can display and forward reporting results using the following options: you can choose a type of output list (basic list, statistics, or ranked list), and you can determine how data is output to the full screen (for example, *SAP List Viewer*, *standard list*, *spreadsheet*, *word processing*). On the InfoSet Query screen, data is always output to SAP List Viewer. Furthermore, the preview of output includes formatting options such as summation and sorting.

See also:

[HR Logical Databases \[Seite 270\]](#)

[InfoSets in the HR Application \[Seite 266\]](#)

Switch Object Selection On/Off

Switch Object Selection On/Off

Use

You are advised to use object selection in InfoSet Query when working with HR InfoSets. The advantage of object selection is that it ensures good selection performance. Furthermore, the following functions can only be used in InfoSet Query if you choose to work with object selection:

- Hit list determination
- Hit list editing
- Set operations
- Restriction of the reporting set

In basis mode, you can make selections according to all of an InfoSet's fields. These options are restricted when object selection is used.






The "*Object selection* switched on" setting is saved as a user setting so that you do not have to enter it time and time again. The next time you use InfoSet Query, this setting is loaded.

Switch Object Selection On

You have started InfoSet Query. The object selection is switched off.




1. Choose *Extras* → *Switch on object selection*.

The object selection is switched on. The additional *Reporting set* group box, and the  *Hit list*,  and  pushbuttons are added to the InfoSet Query screen.

Switch Object Selection Off

You have started InfoSet Query. The object selection is switched on.

1. Choose *Extras* → *Switch off object selection*.

The object selection is switched off. The *Reporting set* group box, and the  *Hit list*,  and  pushbuttons are removed from the InfoSet Query screen.

Overview of Functions

Use

Unlike SAP Query with its broader range of functions, InfoSet Query is also suitable for end users with little experience of HR Reporting. Even beginners are quickly able to use it, thanks to its clear list of available fields and preview of output function.

Prerequisites

The object selection is switched on.



If you start InfoSet Query from the SAP Easy Access menu, an appropriate InfoSet is used automatically in the information systems of several HR components (for example, the information system for Personnel Administration). Provided that you are assigned to the appropriate user group, you can start creating queries immediately. You can also save the created queries, but you cannot transport them using the Workbench Organizer.

If you want to start InfoSet Query in this way, you must be assigned to the appropriate user group. You can set up this assignment in SAP Query (see [Functions for User Group Administration \[Seite 285\]](#)).

Features

- In the top left section of the initial screen, the current InfoSet is displayed with its field groups in an overview tree. Each field group corresponds to an HR infotype. This overview tree enables you to choose selection fields and output fields. If you require further information, see [InfoSet Display \[Seite 18\]](#).



The InfoSet that you use determines the object type that can be selected with InfoSet Query. If you use an InfoSet from Personnel Administration (logical database PNP), for example, the result of a selection is always a set of persons. However, if you use an InfoSet from Personnel Planning, the result of a selection is always a set of objects of a specific object type. The object type, such as business event or position, is determined when the InfoSet is created. If you require further information, see [HR InfoSets for InfoSet Query \[Seite 70\]](#) and [InfoSets in the HR Application \[Seite 266\]](#).

- In the top right section of the screen, the system displays the reporting period, reporting set, list of selection fields and options, and number of hits in the hit list. This screen section is used to determine selection criteria, and execute object selections. You can open and close the *Reporting period* and *Reporting set* screen sections. If you require further information, see [Selection Display \[Seite 23\]](#).
- Once output fields have been selected, a preview of output with example data is displayed in the lower section of the screen. You can format output at this point. For example, you can change the order of columns. Furthermore, this screen section enables you to start outputting real data. If you require further information, see [Output List Display \[Seite 25\]](#).

Overview of Functions

The screenshot displays the SAP Ad-hoc-Query interface. The top menu bar includes Query, Edit, Goto, Extras, System, and Help. Below the menu is a toolbar with various icons. The main window is titled "Ad-hoc-Query" and contains several panels.

Field group/fields Selection:

Field group/fields	Selection	Output
Selection fields from InfoSet		
Actions		1
Personnel Number	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Action Type	<input type="checkbox"/>	<input type="checkbox"/>
Reason for Action	<input type="checkbox"/>	<input type="checkbox"/>
Leaving date	<input type="checkbox"/>	<input type="checkbox"/>
Entry date	<input type="checkbox"/>	<input type="checkbox"/>
Length of service (in year)	<input type="checkbox"/>	<input type="checkbox"/>
Length of service (in mon)	<input type="checkbox"/>	<input type="checkbox"/>
Length of service (in days)	<input type="checkbox"/>	<input type="checkbox"/>
Organizational Assignment	1	1
Company Code	<input type="checkbox"/>	<input type="checkbox"/>
Personnel Area	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Employee Group	<input type="checkbox"/>	<input type="checkbox"/>
Employee Subgroup	<input type="checkbox"/>	<input type="checkbox"/>

Reporting period: Today 02/16/2000

Reporting set: Unrestricted

restrict by: Current hit list

Field name, Option, Value:

Field name	Option	Value
Personnel Area	=	100
Age of Employee	>	35

Hit list: 17 Persons

Data Table:

PA	Personnel Area	Personnel Number	First name	Last name	Kms
200	Personnel Area1	Personnel Number3	First name11	Last name11	14
1900	Personnel Area10	Personnel Number7	First name13	Last name15	2
1100		Personnel Number4	First name7	Last name3	9
2100		Personnel Number3	First name6	Last name8	15
0001		Personnel Number6	First name6	Last name6	15

The bottom status bar shows a green checkmark, a play button, and the text "Q00 (1) (002) pwd".

Activities

The numbering represents the principle of reporting using InfoSet Query with object selection. For detailed instructions, see the specific sections.

1. Selection fields and output fields are chosen from field groups
2. Values and selection options are entered for the selection fields, or a reporting period is selected

3. All of the objects that meet the selection criteria are selected (for example, persons)
4. Output is formatted, and output starts in this screen section if necessary
5. The required data is output to a full screen

See also:

[Selection \[Seite 58\]](#)

[Output \[Seite 66\]](#)

Selection

Selection

Use

When the selection is made, the system determines which objects meet the specified selection criteria in the reporting period. The result of the selection is a set of objects. After the selection has been made, you can continue processing this set or output data on the objects as required.



InfoSet Query only enables you to select one object type. The InfoSet you use determines which object type can be selected. If you use an InfoSet from Personnel Administration, for example, the result of a selection is always a set of persons. InfoSets from Personnel Planning can only be used to select an object type, such as business events or positions. If you require further information, see [HR InfoSets for InfoSet Query \[Seite 70\]](#) and [InfoSets in the HR Application \[Seite 266\]](#).

This section gives you an overview of options available for selecting objects, and continuing to process this set.

Activities

Selection Using One Or More Selection Criteria

In the simplest of cases you can use a selection criterion. But you can use as many selection fields as you want. To do so, choose the selection fields from the overview tree, choose a selection option, and enter a value. You then start the selection. The result of the selection is the objects that meet the selection criteria and those that lie within the specified reporting period (such as all of the business events taking place in New York in March 2001).



You can also make a selection without any selection criteria whatsoever. The result of the selection is all of the objects in the system that can be selected with the InfoSet you have used, and which are valid in the specified reporting period.

Restriction of the Reporting Set

Unless you restrict the reporting set, the selection is made for all of the objects in the system that can be selected with the InfoSet you have used. However, reporting is often required for a specific set of objects only, such as the employees in a particular department or the business events taking place in a particular period. To achieve this aim, you must restrict the reporting set accordingly.

You can restrict the reporting set as follows:

- Using the current hit list, which you obtained by making a specific selection
- Using a set of persons, which you selected using the organizational structure



In addition to the options described here, your system can include other, customer-defined restriction options. If you require further information on the restriction options for the reporting set, access Customizing for the Human Resources Information System and see [Selections IDs \[Extern\]](#), [Define Selection IDs \[Extern\]](#), and [Define Groupings \[Extern\]](#).

If you require more detailed information, see [Restrict the Reporting Set \[Seite 63\]](#).

Processing the Selected Set of Objects

Once the selection has been made, you can continue processing the selected set of objects. You can:

- Display the list of selected objects
- Delete objects from the list
- Branch to HR master data or detail maintenance

If you require further information, see [Process the Selected Set of Objects \[Seite 62\]](#).

Make Selection

Make Selection

Prerequisites

The object selection is switched on.



If you start InfoSet Query from the SAP Easy Access menu, an appropriate InfoSet is used automatically in the systems of several HR components (for example, the system for Personnel Administration). Provided that you are assigned to the appropriate user group, you can start creating queries immediately. You can save the created queries, but you cannot transport them using the Workbench Organizer.

If you want to start InfoSet Query in this way, you must be assigned to the appropriate user group. You can set up this assignment in SAP Query (see [Functions for Managing User Groups \[Seite 285\]](#)).

Procedure

Before working with InfoSet Query for the first time, read the [Overview of Functions \[Seite 55\]](#) to get a basic idea of how InfoSet Query is used.

1. Start InfoSet Query as described in the section [Starting InfoSet Query \[Seite 16\]](#).

This brings you to the InfoSet Query screen.

If you are assigned to a role that supports Reporting with InfoSet Query, and you have started InfoSet Query from the role menu, InfoSet Query starts with an InfoSet, and a query. This means, you can start to determine a new query or to execute or edit the loaded query straight away.

If you are not assigned to a role and have started InfoSet Query from SAP Query or from a menu, you have the following options:


- First you reach a dialog box in which you can select a query area, a user group and an InfoSet. This brings you to the InfoSet Query screen in which the InfoSet Query is displayed, but no selections have been made.
- InfoSet Query is started directly with an InfoSet or a query. You can start working straight away.

2. Expand the field groups you want in the overview tree.
3. Put a tick in the *Selection* column, use drag & drop or choose the context menu to select one or more fields as selection fields.

The system transfers the selection fields to the list of selection fields on the right of the screen. The corresponding value field can be maintained.

4. In the *Value* field, enter selection values for the selection field you have chosen. If necessary, use input help for selecting selection values.

The value types (numeric, alphanumeric, date) that you can enter for the selection fields depend on the type of selection field.

5. In the *Options* field, choose selection options (such as < or =).
6. Choose a reporting period.
7. Choose  *Hit list*.

Result

In the *Hit list* field, the system displays the number of objects that meet your selection criteria.

You can now

- [edit the selected object set \[Seite 62\]](#)
- use the set of objects selected as a reporting set for further selections (see [Restrict the Reporting Set \[Seite 63\]](#))
- output information on the objects selected (see [Output \[Seite 66\]](#)).

Edit Selected Set of Objects

Use





You can display a hit list, which you have generated with InfoSet Query, in a list. You can delete objects from this list and execute a detail display function for individual objects. If you select a set of persons, you branch to HR master data. If you select different objects, you branch to detail maintenance for Personnel Planning. If you remove objects from the set, they are not included in output.

The same editing options are available for all other sets of objects that you can use in InfoSet Query, such as the reporting set and the set of objects for set operations.

Prerequisites

A set of objects exists.

Procedure

1. Choose  for the set that you want to display.
This takes you to the list of objects.
2. To branch to HR master data, select a person and choose *HR master data* (this is only possible if you select a person). To branch to detail maintenance, select an object and choose *Detail maintenance* (only if you selected objects from Personnel Planning)
3. To delete an object from the list (because you do not want it to be included in the following report, for example), select the object and choose .
4. To accept changes, choose . To cancel changes, choose .

Result

You go back to the InfoSet Query screen. If you deleted objects, the set of objects is reduced by the corresponding number.

Restrict the Reporting Set

Use


Restrict the reporting set if you only want to report on a specific set of objects, such as a specific group of persons.



In addition to the options described here, your system can include other, customer-defined restriction options. If you require further information on the restriction options for the reporting set, access Customizing for the Human Resources Information System and see [Selection IDs \[Extern\]](#), [Define Selection IDs \[Extern\]](#), and [Define Groupings \[Extern\]](#).

Procedure (Restriction Using Organizational Structure)

You use an InfoSet from Personnel Administration, that is, an InfoSet based on logical database PNP.

1. Under *Restrict reporting set by*, choose the *Organizational structure* entry.
2. To access the organizational structure, click on the *Organizational structure* entry.
3. Select the required organizational unit(s). You can select one or more organizational units, entire branches of an organizational structure, or a complete organizational structure.
4. Confirm by choosing .

The persons who belong to the selected organizational units are transferred as the reporting set. The number of persons is displayed in the *Reporting set* field.



If you make your selection by using the organizational structure, the system selects the persons who belong to the selected organizational unit or one of the underlying organizational units. However, if you use the *Organizational unit* as a selection field, you must enter the upper-level and underlying organizational units as values to select the same hit list.

Procedure (Restriction Using Current Hit List)

You use an InfoSet as required.

1. Make your selection as described in [Make Selection \[Seite 60\]](#).
2. Under *Restrict reporting set by*, choose the *Hit list* entry.
3. To transfer the current hit list as a reporting set, click on the *Hit list* entry.

The current hit list is transferred as a reporting set, and can be used for further selections. The number of objects is displayed in the *Reporting set* field.

Set Operations

Use

You can use set operations to relate two sets of objects to each other. You can use the resulting set for output, as a reporting set for further selections, or for further set operations.

Prerequisites

You are working with object selection.

Features

You can show the *Set operations* tab page for the set operations. The set operations process the sets from the *Object set A* and *Object set B* fields, and write the result to the *Resulting set* field.

You can perform one of the following set operations:

- Create intersection
- Create union
- Set A minus set B
- Set B minus set A

If you want to use the resulting set for output, you must copy it to the *Hit list*. If you want to use the resulting set as a reporting set for further selections, you must copy it to the *Reporting set*. If you want to use the resulting set for further set operations, you must copy it to set A or set B.

If you require further information, see [Execute Set Operations \[Seite 65\]](#).

Example

You want to run a report to determine which employees at your enterprise do not have a temporary residence. You cannot select non-existent data records, so you implement the report according to the following logic:

- First, you select all of the employees at your enterprise and copy this set to *Object set A*.
- You select all of the employees with a temporary residence and copy them to *Object set B*.
- Set operation *Set A minus set B* enables you to subtract all of the persons with a temporary residence from the set of all employees. The resulting set contains all of the persons who do not have a temporary residence.

Execute Set Operations

Use

You can use set operations to relate two selection results (create intersection, create union, subtract sets).

Prerequisites



You are working with object selection. You have made a selection, and the system has displayed a hit list.




The "Set operations shown" setting is saved as a user setting so that you do not have to enter it time and time again. The next time you use InfoSet Query, this setting is loaded.

Procedure


1. Choose *Extras* → *Show set operations*.

The additional *Set operations* tab page is provided by the system. Both of the additional  and  pushbuttons are added to the *Selection* tab page.

2. On the *Selection* tab page, choose .

The hit list is copied to the *Object set A* field.


3. Make a second selection.

4. On the *Selection* tab page, choose .

The hit list is copied to the *Object set B* field.


5. Choose the *Set operations* tab page.

The *Object set A* field contains the hit list from your first selection. The *Object set B* field contains the hit list from your second selection.

6. To perform the required set operation, select it and then confirm by choosing  *Perform operation*.

The *Resulting set* field contains the result of the set operation. You can now process the resulting set as follows:

- Copy to the hit list for output
- Copy to *Object set A* or *Object set B* for further set operations
- Copy to the reporting set, and then use it for further selections

7. To copy the resulting set, select one of the transfer options for the resulting set and choose  *Copy resulting set*.

Result

The resulting set is transferred to the specified set, where you can continue to process it.

Output

Output

Use

InfoSet Query enables you to output data on selected objects as required. You can start output directly on the InfoSet Query screen, or as a full screen. On the InfoSet Query screen, the results list is output to SAP List Viewer. If you want to output data to a full screen, you can choose between different types of output.

Features

You can choose a type of output list for output. This determines if and how data is aggregated when output. You can select one of the following options:

- Basic list
- Statistics
- Ranked list

You can also choose the type of output. This determines how data is displayed when output:

- SAP List Viewer
- Standard list
- Table control
- Word processing
- Spreadsheet
- Private folder
- Graphic
- Crystal Reports
- Info Zoom

You can also format output in the preview of output. This enables you to determine how data is arranged when output (for example, the sequence of columns and output list sorting), and whether numerical columns are totaled.

Activities

1. You determine the *settings* (see [Settings \[Seite 39\]](#))
 - Which type of output list is used
 - Which type of output is used for outputting data to a full screen
 - Whether a selection screen is displayed before output
 - Whether the selection conditions are taken into account when data is output
 - How many ranks are output to a ranked list
 - How data is exported to Crystal Reports
 - Whether the value or text is used when output fields are selected using drag & drop or checkboxes
2. You select output fields from the overview tree by selecting the appropriate checkboxes in the *Output* column, by using drag & drop, or by using the appropriate context menu. You can determine the output fields at the same time as the selection fields, or after you have made your selection. You can choose output fields from the field groups as required.



You can output all data on the selected persons as a *Value*, and you can also output some of the data as a *Text*. For example, the *form-of-address key*: value: 01, text: Mr; value: 02, text: Ms.

The icon used to flag a field indicates whether the field has a text. The context menu for each field with a text enables you to determine whether the text, the value, or both are output. If you select fields by using the checkboxes or drag & drop, the standards determined under *Settings* apply.

3. You use the preview of output to format output as required (see [Format Output \[Seite 27\]](#)).
4. You start output in the InfoSet Query screen, or as a full screen.



Output Data for Selected Objects

Prerequisites

Maybe you have made a selection and edited the hit list. If you do not want to edit the hit list, you can start output without a prior selection. The selection is made automatically before data is output.

You are advised to use data output after you have entered selection criteria. It is possible to start output without selection criteria, but this means that all of the objects that can be selected using the current InfoSet are indeed selected and output. This can result in extremely long runtimes.

Procedure

1. If necessary, determine the output settings (see [Settings \[Seite 39\]](#)).
2. In the overview tree, navigate to the required field groups and fields.
3. Select the required output fields by setting flags in the *Output* column.
You can also determine output fields by drag & drop. To do so, select one or more fields and drag them to the preview of output.
You can also determine output fields using the context menu. To do so, select a field and use the right mouse button to access the context menu. In the *Output* context menu, select one of the output options. (If a field has a text, the options are *Text only*, *Value only*, and *Value and text*. If a field does not have a text, the option is *Value only*.)
The system transfers the fields to the preview of output as columns. If fields have already been selected for output, the new columns are added to the list after the selected fields.
4. Format output as required (by changing the order of columns, for example; see [Format Output \[Seite 27\]](#)).
5. To start output in the InfoSet Query screen, choose  from the application toolbar in the preview of output.
6. To start output in a full screen, choose  *Output*.

Result

The information on the selected objects is output in accordance with your entries.

In the standard system, the data records that meet the selection conditions are output for each selected object. However, if you want to output all of the records that exist for each selected object, choose *Edit* → *Settings*, select the *Output* tab page, and deselect the *Use selection conditions* indicator.



In the first step, you select all persons whose permanent residence is New York. The result is the set of all persons whose permanent residence is New York.

If the *Use selection conditions* indicator has been set, all records are output whose permanent residence is New York. If the indicator has not been set, all other places of residence are also output for the selected persons (for example, Tokyo and Munich as well as New York).

Process Selection Result in General Reporting

Use

You can continue to process the selected set of objects with standard reports that are in the same logical database as the InfoSet used in InfoSet Query, and which process the same object type that you selected in InfoSet Query.

Procedure


1. Make a selection (see [Make Selection \[Seite 60\]](#)).

2. Choose *Goto* → *Start report*.

This takes you to a dialog box in which you can enter a report name, or choose a report using input help.



You can only process the set of objects in reports that are based on the same logical database as the InfoSet used. Furthermore, you can only use reports that can process the same object type as the one you selected in InfoSet Query.

3. Enter a report name in the *Report* field, or choose a report.
4. If you want to carry out the report with variants, choose a *Variant*.
5. If you want to start the report via the selection screen so you can restrict particular values, for example, you must select the *Start via selection screen* indicator.
6. Choose .

Result

The report you have chosen is executed for the selected objects.

HR InfoSets for InfoSet Query

To access data stored in the system, InfoSet Query uses an InfoSet. It provides you with a view of data in specific parts of HR that is structured by infotype. On the initial InfoSet Query screen, the InfoSet is displayed with its field groups as an overview tree. This overview tree enables you to choose selection fields and output fields.

The InfoSets used in InfoSet Query are created and managed in SAP Query. When you create an InfoSet, you select a logical database on which the InfoSet is based, and determine which infotypes are included in the InfoSet. They are displayed in the InfoSet as field groups. After you have selected the infotypes, you can determine which fields are included in the field groups for each infotype.



In the standard system, all infotype-specific fields of the selected infotypes are included in the InfoSet. You are advised to check the selection of fields carefully, and adapt the selection to your reporting concept. Appropriately adjusted InfoSets are user-friendly, which makes working with InfoSet Query much simpler.

The InfoSet determines the objects that you can select with InfoSet Query. The following scenarios are possible:

- The InfoSet is based on logical database **PNP**

InfoSets based on this logical database enable you to use InfoSet Query to select **employees**. You can use data from Personnel Administration and Time Management and payroll infotypes as selection criteria. From a technical perspective, this means you can use fields from infotypes 0000 to 0999 and 2000 to 2999 and payroll infotypes as selection criteria. For example, you can use the InfoSet to run a report that determines which employees have a particular place of residence.

Furthermore, you can create InfoSets on the basis of this logical database that enable you to report on the infotypes of related objects using InfoSet Query. For example, you can select persons who participated in a particular business event, and output the qualifications of the persons selected. To do this, you must select infotypes from Personnel Planning as well as infotypes from Personnel Administration when you create the InfoSet.

- The InfoSet is based on logical database **PAP**

InfoSets based on this database enable you to use InfoSet Query to select **applicants**. You can use data from Recruitment as selection criteria and for output. From a technical perspective, this means you can use fields from specific Personnel Administration infotypes (such as 0001 and 0002) and fields from infotypes 4000 to 4999 as selection and output fields.

- The InfoSet is based on logical database **PCH**

Provided that object selection is switched on, InfoSets based on this database enable you to use InfoSet Query to select **objects of one object type**, such as business events, qualifications, and positions. You can use all of the fields of infotypes allowed for the object in question, and all of the object types and their allowed infotypes that can be related with the selected object type, as selection criteria and for output.

When you create an InfoSet, you determine the object type that you can select using an InfoSet. The generated InfoSet can only be used to select this particular object type. If you want to create reports for business events, qualifications, or positions, for example, you must create three separate InfoSets. This means the reports can only be executed separately.

If you do not select an object type when you create the InfoSet, you can only use the InfoSet for InfoSet Query if object selection has been switched off.

See also:

[InfoSets in the HR Application \[Seite 266\]](#)

SAP Query

Purpose

The application SAP Query is used to create lists not already contained in the SAP standard. It has been designed for users with little or no knowledge of the SAP programming language ABAP. SAP Query offers users a broad range of ways to define reporting programs and create different types of reports such as basic lists, statistics, and ranked lists.

Features

SAP Query's range of functions correspond to the classical reporting functions available in the system. Needs in this area like list, statistic, or ranked list creation can be met using queries.

All necessary data can be selected from various R/3 tables.

To define a report, you first have to enter individual texts, such as titles, and select the fields and options which determine the report layout. Then you can edit list display in WYSIWYG mode whenever you want using drag and drop and the other toolbox functions available.

List of Components

SAP Query consists of four components. The following overview lists these components and the menu options you select to access them.

- Query Maintenance:
Tools → ABAP Workbench → Utilities → SAP Query → Queries
- Maintaining InfoSets:
Tools → ABAP Workbench → Utilities → SAP Query → InfoSets
- User Group Maintenance:
Tools → ABAP Workbench → Utilities → SAP Query → User Groups
- Language Comparison (Translation):
Tools → ABAP Workbench → Utilities → Translation → SAP Query

Each screen of each component includes the *Environment* option on the menu bar. Any of the other components can be accessed via this option.

Groups of Query Users

Groups of Query Users

SAP Query is used by three different groups of people. Each of these groups works with different SAP Query components.

- Departmental users
These are end-users who create new queries and generate lists by executing queries.
- System administrators
System administrators set up the necessary environment for end users and carry out transports.
- Translators
Translators perform a language comparison for the different texts.

Maintaining Queries

The component *Maintain Queries* is aimed at end users who want to define and modify queries, as well as generate lists by executing queries.

Maintaining User Groups

System administrators use the component *Maintain User Groups* to set up the appropriate environment for end-users. Users working within the same application are thus assigned to the same user group.

Within a user group, it is irrelevant who has defined a certain query, since anyone who belonging to that group can execute it. However, only users with the appropriate authorization can change queries or define new ones. Users are not allowed to modify queries from other user groups, although they may, under certain circumstances, copy and execute them (see [Copying Queries \[Seite 98\]](#)). Each user can be assigned to several user groups.

Maintaining InfoSets

System administrators use the component *Maintain InfoSets* to set up the appropriate working environment for end users.

InfoSets provide special views of data sources, i.e. they determine which fields of a data source can be evaluated in queries. InfoSets are assigned to user groups.

By creating InfoSets and assigning them to user groups, the system administrator determines the range of reports the individual application departments or end-users can generate using SAP Query.

A certain number of InfoSets are available to an end-user for each user group of which he is a member. These InfoSets are those applicable to the work area that is characterized by the user group. An InfoSet can be assigned to several user groups.



Vendor master data can be relevant for Purchasing as well as for Accountancy. The corresponding InfoSet is then assigned to the two user groups. This means that queries based on this InfoSet can be copied and executed by both groups.

Query Areas

Query Areas

A query area contains a set of query objects (queries, InfoSets, and user groups) that are discrete and consistent.

You can differentiate between two different query areas, the standard area and the global area

- **Standard Area**

The standard query area is primarily designed for 'ad-hoc queries,' that is for queries that are created to fulfill a one-time demand and never used again. Standard area queries are client-specific. They are not connected to the Workbench Organizer.

- **Global Area**

In the global query area, queries are developed that are used throughout the entire system (that is, they are cross-client). These queries are also intended for transport into other systems.

Query objects created in the global area are registered in the Workbench Organizer. They can be created and transported using the normal correction and transport procedure.

All query objects delivered by SAP (from Release 4.0) are located in the global area. You can tell which objects have been delivered by SAP and which objects are customers objects by looking at their prefixes (see [Query Areas \[Seite 297\]](#)).

Both query areas provide you with a full range SAP Query functions.

Transports

- Global Area

Global area objects are created and transported via the Workbench Organizer.

- Standard Area

The components *Maintain InfoSets* and *Maintain User Groups* contain utilities for transporting standard area objects (user groups, InfoSets and Queries). This enables you, for example, to transfer tested queries to the production system.

Language Comparison

Defining queries, InfoSets and user groups involves entering a lot of text. Some of this contributes to the internal organization of SAP Query, while the rest is used in the generated ABAP programs.

A language comparison facility exists for all these text elements. This means that, for each text, there is an equivalent text in one (or more) other languages. The texts you see in your SAP R/3 System are usually displayed in the logon language.

You perform this language comparison with the component *Language Comparison*.

SAP Query Authorizations

End users, system administrators, and translators must all have appropriate authorizations to use SAP Query. For example, it would not make sense for end-users to have the authorization to maintain InfoSets.

Also, it can be desirable to set up authorizations within a user group in such a way that some end-users can maintain and execute queries, while others can only execute existing queries.

SAP Query has two means of assigning authorizations to individuals:

- User groups

To be able to use the component *Maintain Queries*, you must be a member of at least one user group. In all other user groups, maintenance may only be performed after your user name has been explicitly assigned to them. This means that you can only access certain InfoSets.

- Authorizations

You can also assign authorizations using the authorization object S_QUERY.

This authorization object contains a field ACTVT, which can accept the values **Change** (02), **Maintain** (23) and **Translate** (67). You can assign authorizations for this authorization object.

Authorizations for this object always refer to both work areas. If a user has authorization to change queries, this means that he or she can initially create and change queries in all user groups within the standard and global areas for which he or she has been entered.

Action:	Authorization:
Maintaining Queries	<p>To be able to create new queries or modify existing ones in the component <i>Maintain Queries</i>, users must have an authorization for the authorization object S_QUERY with the value Change (02). This authorization must also be confirmed for the corresponding user group, that is, not revoked.</p> <p>Authorization to change objects can, however, in addition be explicitly restricted to individual user groups (see Assigning Users and InfoSets [Seite 290]).</p>
Executing Queries	<p>If a query accesses a certain table when it is run, the user needs display authorization for authorization object S_TABU_DIS. Field DICBERCLS must contain the table's authorization groups.</p> <p>This authorization object protects all tables from unauthorized access. If you are accessing tables that are part of a logical database, authorization for data access can be set up using the logical database. Further information can be found under Logical Databases [Extern]</p> <p>This is the same authorization that you need in order to be able to display tables using either the Data Browser (transaction SE16) or the initial table maintenance screen (transaction SM31).</p>

SAP Query Authorizations

Maintaining InfoSets	<p>The component <i>Maintaining InfoSets</i> can only be accessed by users with authorization for the authorization object S_QUERY and the appropriate value for Maintenance (23)</p> <p>The authorization for maintaining InfoSets is restricted in such a way that a user wanting to store some ABAP code in an InfoSet can do this only if s/he has authorization for maintaining the authorization object S_DEVELOP with value 'PROG' for field OBJTYPE and with value 'AQ*' for field OBJNAME.</p> <p>This is the same authorization that you need in order to be able to use the ABAP Editor to create or change programs whose names begin with 'AQ'.</p> <p>If s/he does not have this authorization, then s/he can only select fields, connect additional tables or structures and define parameters and selection criteria.</p>
Maintaining User Groups	<p>The component <i>Maintain User Groups</i> can only be accessed by users with authorization for the authorization object S_QUERY and the appropriate value for Maintenance (23)</p>
Language Comparison	<p>The component <i>Language Comparison</i> can only be accessed by users with authorization for the authorization object S_QUERY and the appropriate value for Translation (67).</p>

Users who have authorization for the authorization object S_QUERY with both the values **Change** and **Maintain**, can access all queries of all user groups without being explicitly entered in each user group.

Release Notes: Changes to SAP Query

All changes and enhancements to SAP Query are detailed in the Release Notes documentation. Choose *Help* → *Release notes* from any screen.

Application

Application

The following sections are directed towards the end user: This chapter explains how to administer, run, create, and change queries.

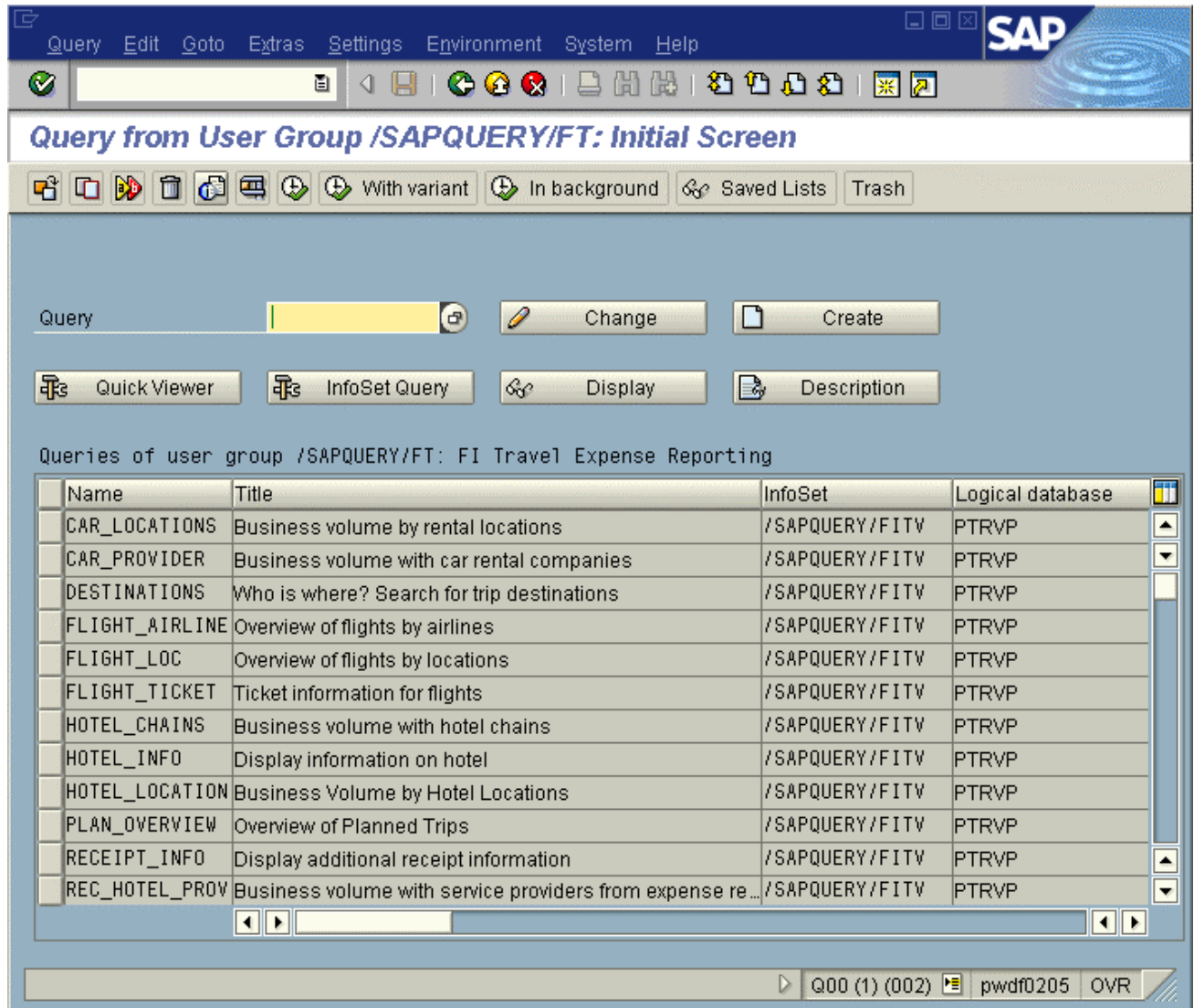
Functions for Managing Queries

To manage queries, you use the component *Maintain Queries*. For further information, refer to the following sections.

Choosing a Query for Execution or Maintenance

Choosing a Query for Execution or Maintenance

The initial screen of the component *Maintain Queries* looks similar to the one below.



If you are not authorized to change queries, the *Change* and *Create* functions do not appear on this screen. You can, however, display a query at any time.

When you start the component, the system takes a user group to which you are assigned and displays the names and titles of any queries already defined for this user group in the lower part of the screen. By selecting *Other user group*, you can display the user groups to which you are assigned and choose one for further processing.

A range of display and maintenance functions is available to you for manipulating these queries.

To select a query for processing by one of these functions, proceed in either of the following ways:

- Enter the name of the query in the appropriate input field, or

Choosing a Query for Execution or Maintenance

- Select the list entry for the query on the lower part of the screen.

You can now execute the function.



All the examples in this manual are based on a flight reservation system. The data for this system is stored in a number of tables and in a logical database. (see *Logical Database F1S - Evaluating Flight Bookings* in the appendix)

Displaying Directories

The following sections explain how to display information about user groups, queries, and InfoSets:

User Group Directory

To display a directory of the user groups to which you are assigned, select *Other user group*. You then see a list of two-character user group names together with their long text descriptions. The long text is intended to help you choose a user group:

Name	User group name
/SAPQUERY/AM	Asset manager
/SAPQUERY/AU	Audit
/SAPQUERY/DL	Service
/SAPQUERY/FB	FI Document Analysis
/SAPQUERY/FD	FI Customer Analysis
/SAPQUERY/FK	FI Vendor Analysis
/SAPQUERY/FP	FI General Ledger Evaluations
/SAPQUERY/FS	FI General Ledger Analysis

You can scroll through this display and choose a user group. The system subsequently returns to the initial screen and displays the chosen user group in the lower part of the screen. You can then choose a query you want to execute or edit.

Directory of Queries

Directory of Queries

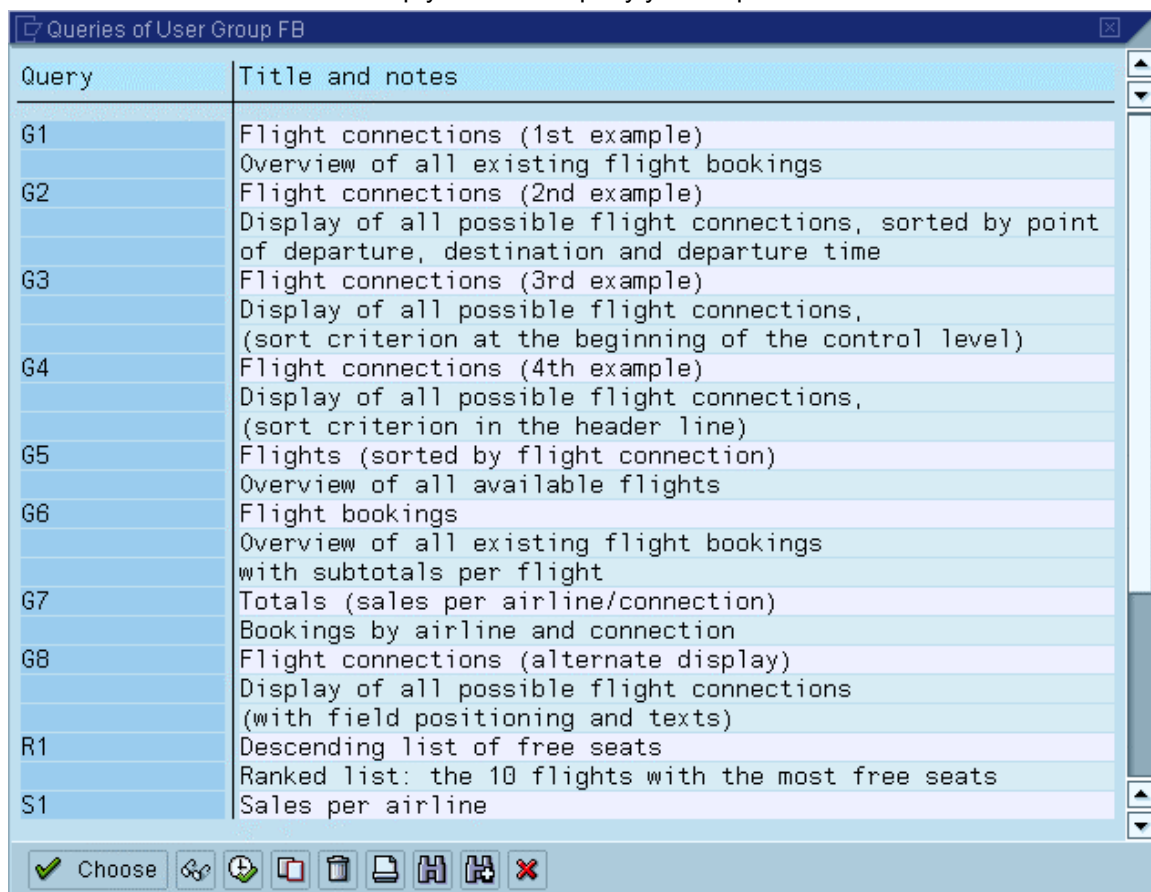
When you have chosen a user group, you see the names and titles of any queries already defined there in the lower part of the screen.

Directory of Queries for a User Group

If you need more information before selecting a query, you can display a list of queries for your current user group. To do this, proceed as follows:

- Select *Environment* → *Directories* → *Queries/user group*.
- Place the cursor on the input field for the query name and display a list of possible entries.

For each query, the resulting list contains the name, the title and up to three lines of notes. The title and notes are intended to help you find the query you require.



Query	Title and notes
G1	Flight connections (1st example) Overview of all existing flight bookings
G2	Flight connections (2nd example) Display of all possible flight connections, sorted by point of departure, destination and departure time
G3	Flight connections (3rd example) Display of all possible flight connections, (sort criterion at the beginning of the control level)
G4	Flight connections (4th example) Display of all possible flight connections, (sort criterion in the header line)
G5	Flights (sorted by flight connection) Overview of all available flights
G6	Flight bookings Overview of all existing flight bookings with subtotals per flight
G7	Totals (sales per airline/connection) Bookings by airline and connection
G8	Flight connections (alternate display) Display of all possible flight connections (with field positioning and texts)
R1	Descending list of free seats Ranked list: the 10 flights with the most free seats
S1	Sales per airline

You can scroll through this display. You can also place the cursor on a query name and call one of the functions offered in the bottom part of the dialog window for the chosen query.

You can use the *Delete* function only if you have change authorization for queries.

Directory of All Queries

If you want a list of queries for all user groups, choose *Environment* → *Directories* → *All queries*. However, you should be aware that the system displays only the queries which your current user group is allowed to copy and execute. In this directory, there is one page for each user group:

Query	: G9	Flight bookings
User group	: FB	Flight bookings
General specifications		
Author	ABAP	31.08.1999
Last changed by	ABAP	31.08.1999
Notes		
Overview of all existing flight bookings with subtotals per flight		
Functional area		
FLBU	Flight bookings	
Origin of data		
Logical database F1S		
Generated lists		
Additional selections		
None		
Standard variant		
None		

You can also scroll through this display. You can position the cursor on a query and call one of the functions *Choose*, *Description*, *Execute* (and so on), for the chosen query.

You can use the *Copy* function only if you have a change authorization for queries.



The list of all queries is often very long. If, therefore, you are interested in the queries of one particular user group, you can use the *user groups* function to choose a user group and display the associated queries.

InfoSet Directory

InfoSet Directory

If you have authorization to change queries, you can display a list of all InfoSets which are assigned to the user group you have chosen. You are allowed to create queries based on these InfoSets.

To display the directory, choose *Environment* → *Directories* → *InfoSets*.

Managing Queries

The following sections describe what kinds of query management functions are available to the user.

Displaying Queries

To display information about the structure of a query, first enter the user group and the name of the specific query you are interested in. Now you can either choose *Description* or *Display*. If you want to see all of the details of the query's definition, choose *Display*. All screens are displayed in the same manner as if the query were being changed, however no new entries can be made.

If you are interested in an overview of the query, without wanting details about its list layout, choose *Description*.

First, you see an overview containing general information about a query. This includes:

- Title of the query
- Author and last person to make changes
- Notes about the query
- InfoSet used to define the query
- Origin of the selected data
- Generated list types (sublists)
- Additional selections

Information about individual sublists is also displayed on the screen. If the query contains a combination of these, the basic list is displayed first, followed by the statistics and, finally, by any ranked lists. For each sublist, the information specifies which fields are output in which order, and the options used to achieve this.

Description of basic lists:

Query	: G1	Flight connections (1st example)
User group	: FB	Flight bookings

Basic list

Displayed fields

Line	No.	Field
01	01	From city
	02	Destination
	03	Airline carrier Id
	04	Flight connection Id
	05	Departure time
	06	Arrival time

Sort sequence

None

Summation

None

Inventory count

None

Description of statistics:

Displaying Queries

Query	: S2	Sales per airline and flight connection			
User group	: FB	Flight bookings			
Statistic no. 1 Sales per airline and flight connection					
Displayed fields					
No.	Field	Format			
		Number	Average	%	Round
01	Airline carrier Id				
02	Flight connection Id				
03	Total of current bookings	X	X	X	
	Reference unit: DEM				
Sort sequence					
Level	Sort field				
01	Airline carrier Id (ascending)				
02	Flight connection Id (ascending)				

Description of ranked lists:

Query	: R1	Descending list of free seats			
User group	: FB	Flight bookings			
Ranked list no. 1 Free seats per flight					
The following are output: 10 Seats					
Displayed fields					
No.	Field	Ranked list criterion			
01	Airline carrier Id	X (descending)			
02	Flight connection Id				
03	Flight date				
04	Free seats				
05	Maximum capacity				

You can call the *Description* function from the query directory (to display this, select *Environment* → *Directories* → *All queries* or *Queries/user group*).



If you select *Layout display*, the system displays the basic layout of the generated list. This function is of particular interest if you are maintaining queries. You can call this function anywhere within the component *Maintain Queries*.

Copying Queries

Copying Queries

You can copy queries only if you have the authorization to make changes. Within your current user group, you can copy all queries. However, queries of other user groups can only be copied if the InfoSet used to define the query is assigned to both user groups.

To copy a query, proceed as follows:

1. Choose the name of the query you want to copy on the initial screen.
If you do not know the name, use the directory functions to display the query directories and then choose a query to copy from there.
2. Choose *Copy*.
3. Enter the name and the user group of the query that you want to copy in the dialog box. Furthermore, you must enter a name for the copied query. The system proposes values for this.

	Query	User group
From	G1	FB
To		FB

4. Choose *Continue*.

This takes you to the initial screen. The query is added and appears in the query directory. You can now continue.



You can also copy queries from the query directory by selecting *Environment* → *Directories* → *All queries*. To do this, place the cursor on the desired query and select *Copy*. In the dialog box, enter the name of the copy.

Renaming Queries

To rename a query, proceed as follows:

1. Choose the name of the query you want to rename on the initial screen.
2. Choose *Rename*.
3. Enter the new name to be given to the selected query in the dialog box.
4. Choose *Continue*.

This takes you to the initial screen. The query has been given a new name and will appear in the query directory.

The following objects have to be locked when a query is being renamed:

- The query catalog of the user group affected
- The query itself
- The query list catalog.

The function is executed only if all these were able to be set. As long as these locks are set, other users may be prevented from continuing their work.

Deleting Queries

Deleting Queries

You can delete queries only if you have the authorization to make changes. To delete a query, proceed as follows:

1. Choose the name of the query you want to delete on the initial screen.
2. Choose *Delete*.

You then see a dialog box where you have to decide once again whether you want to delete the query or not. If you confirm the deletion, the system deletes the query and outputs an appropriate message.



You can also delete queries from the query directory. To do this, place the cursor on the desired query and choose *Delete*. Then, follow the steps described above.

Queries and InfoSets are placed in a temporary storage buffer (trash can) when you delete them. You can retrieve them from there when you want. Every deleted object remains in the trash can for 30 days. After 30 days, all items in the trash can are deleted automatically.

The trash can is administered for each user group. If at least one query from a user group has been deleted, a *Trash can* pushbutton appears in the application tool on its initial screen. When you execute the *Trash* function, a list is displayed of all queries from that user group that have been deleted. From this list it is then possible to select a query with your cursor and either *Retrieve* the query or *Delete* it.

The *Retrieve* function reads the query from the trash and places it back in the user group. The system checks whether the retrieval is possible. If, for example, the corresponding InfoSet has been deleted, retrieval is not possible, at least not until the deleted InfoSet has also been retrieved from the trash. Successful retrieval is documented in a log.

The *Delete* function allows you to permanently delete a query from the trash before thirty days are up.

The *Retrieve* and *Delete* functions both lead to a query's subsequent removal (disappearance) from the trash.

Comparing Queries

Sometimes, differences in the definition of individual fields may occur between a query and the associated InfoSet.



Suppose you have created a query using an InfoSet and have already executed it frequently. Your system administrator then modifies the InfoSet by changing the type of a field which you use in the query. When you next want to use the query, you get a message informing you that differences exist between the query and the InfoSet.

If differences occur between queries and InfoSets, you should terminate the processing and perform a comparison between the query and the InfoSet. This is possible only if you are authorized to change queries.

To perform a comparison, select *Query* → *More functions* → *Compare*. You then see a list of all the fields defined differently, together with information about how the field definition has changed and where in the query the field is used. The system gives you an automatic adjustment option for individual fields. At the end of the list, there are some notes telling you what to look out for in an automatic adjustment and what you must do, if you want to carry out the adjustment yourself.



Differences between queries and InfoSets can also occur after transports.

Executing Queries

Executing Queries

When you have chosen a query and a user group on the initial screen, you can execute the query online or in the background. Further information about this is contained in the following sections.

Executing Queries Online

To execute a query online, select *Query* followed by *Execute* or *Execute with variant*. If you select *Execute with variant*, you see a dialog box where you can enter the variant name.



A variant is a saved set of selection criteria for a query or the report generated by the query. If you specify a variant when starting a query, the system uses the values in the variant for the query selection criteria.

You enter variants on the selection screen and save them under any name. For each query, you can create any number of variants. To create a variant, select *Goto* → *Maintain variants*.

With queries from the global query area you can create system variants. For further information, see [Query Areas \[Seite 297\]](#).

For further information about variants, please refer to the appropriate sections in the *ABAP User Manual* or to the online documentation *Getting started with R/3*.

You can create one standard variant per query. This standard variant is always used when you choose *Execute*. If you select *Exec. with variant*, the system sets the standard variant as the default value (see [Assigning Title, Format, and Notes \[Seite 139\]](#)).

In addition to the functions *Execute* and *Execute with variant*, the function *Test* is also available on query maintenance screens. This function allows you to determine the maximum number of read accesses to the database and thus is especially useful for test purposes.

When you execute this function, a window appears where you can determine the maximum number of data records you want to read. Each individual record is considered a database access. If you are using a logical database to select data from multiple tables, all records from all tables used are counted. The system terminates data selection when the maximum number of selections has been reached. You can also enter a variant in this same window if you want to execute with variant.

After you choose one of the functions described above, the selection screen is sent. Here you may enter selection criteria, just as if you were starting a report. Below is an example of a selection screen.

Executing Queries Online

Customer name		to		
Carrier		to		
Flight date		to		
Order date		to		

Program selections

Output format

☒ SAP List Viewer
☐ ABAP list
☐ Graphics
☐ ABC analysis
☐ Executive Info System (EIS)
☐ File store
☐ Save with ID
☐ Display as table
☐ Word processing
☐ Spreadsheet
☐ Private file

Selection screens normally consist of two parts:

- The upper part of the screen contains the database selections. These are determined by the underlying logical database and are automatically displayed on the screen.
- The lower part of the screen contains program parameters that come either from defining the InfoSet or the query, or that were automatically generated by the query.



These program parameters may consist of any of the following:

- Parameters and selection criteria defined in the InfoSet used.
- Parameters used to define the values of local fields (see [Defining Local Fields \[Seite 145\]](#)).
- Additional selection criteria specified when defining the query.
- A parameter for the currency conversion date, if required by the query.
- Several radio buttons allowing you to send the list for further processing (see [Interactive Functions for Further List Processing \[Seite 117\]](#)).



The radio button SAP List Viewer is set in the selection screen under *Output type*. Many of the query-specific output functions mentioned in the following sections are available to you if you choose *ABAP List* as an output form.

When you have entered the selections, choose *Execute* to display your list.

Overview of flights by airlines							
ID	Flight	Departure	DepartTime	From	Departure Location	to	Flight arrival location
AC	0482	26.01.2000	06:30:00	YYZ	Toronto, Pearson Intl	YUL	Montreal, Dorval
	0137	27.01.2000	16:00:00	YUL	Montreal, Dorval	YYZ	Toronto, Pearson Intl
	0482	28.01.2000	06:30:00	YYZ	Toronto, Pearson Intl	YUL	Montreal, Dorval
	0137	29.01.2000	16:00:00	YUL	Montreal, Dorval	YYZ	Toronto, Pearson Intl
	0400	30.01.2000	07:00:00	YYZ	Toronto, Pearson Intl	YUL	Montreal, Dorval
	0137	01.02.2000	16:00:00	YUL	Montreal, Dorval	YYZ	Toronto, Pearson Intl
	0140	03.02.2000	16:30:00	YYC	Calgary	YYZ	Toronto, Pearson Intl
LH	6502	20.01.2000	08:10:00	FRA	Frankfurt	ORD	Chicago, O'hare Intl
SK	0944	24.01.2000	22:00:00	ORD	Chicago, O'hare Intl	CPH	Copenhagen
	0633	25.01.2000	14:25:00	CPH	Copenhagen	FRA	Frankfurt
UA	3383	02.02.2000	10:00:00	YYZ	Toronto, Pearson Intl	YYC	Calgary
	3521	03.03.2000	10:20:00	FRA	Frankfurt	LAX	Los Angeles
	3373	02.08.2000	11:00:00	YYZ	Toronto, Pearson Intl	YVR	Vancouver
	3378	02.09.2000	22:45:00	YVR	Vancouver	YYZ	Toronto, Pearson Intl
	3375	02.10.2000	13:00:00	YYZ	Toronto, Pearson Intl	YVR	Vancouver
	3378	02.11.2000	22:45:00	YVR	Vancouver	YYZ	Toronto, Pearson Intl

Whenever a query is started and no entries are made on the selection screen, a dialog box appears asking you to limit the number of database accesses. Here you can either enter a number of your choice or confirm the default value (100). This dialog box prevents you from starting queries and mistakenly reading the entire dataset contained within an InfoSet.

All selection criteria and parameters from your selection screen are checked to see if the number of database accesses should be limited. The following parameters are NOT checked:

- all parameters automatically added to the selection screen by the query (date of currency conversion, direct interaction specifications, and so on).
- all invisible parameters, and
- all parameters that appear as either radio buttons or checkboxes on the selection screen.

The query is executed without further testing as soon as the system comes across a single parameter or selection criteria in any of the other fields. If this is not the case, the dialog box asking you to restrict the number of database accesses appears, much the same as the dialog box that appears during the *Test* function. (Exception: this does not occur if you are executing the query with variants or in the background).

If you want to output the list to a printer instead of the screen, choose *Program → Execute and Print* from your selection screen. You then see a screen where you can specify the print options.

For a description of the different options, refer to the *ABAP User Manual* and the online documentation *Getting started with R/3*.



You can also display the list on the screen first and then select *Print*.

If a query contains multiple sublists, a dialog box appears with checkboxes allowing you to select which sublists you want to be printed.

Executing Queries in the Background

If you want to execute the query as a background job, choose *Exec. in background*.

When scheduling a job, you must define a variant and specify the start time.



Please remember that background processing always requires a variant. If you have still not created a variant for the query, you can do this on the initial screen for background processing.

For information on all other options and processes, refer to the online documentation *Getting started with R/3*.

Improving Response Times

Response times depend largely on the type of system you have and the system load at the time, although the number of database accesses needed to process your query also plays a part. The actual processing of the data is virtually irrelevant as far as the runtime is concerned.

Reducing Database Accesses

The InfoSet you choose (see [Creating and Changing Queries \[Seite 133\]](#)) determines the database to be evaluated. Each database has an associated selection screen which is automatically displayed when you start a query. Any selections you enter on this screen directly affect the response time. Therefore, the more precisely you select your data, the shorter the response time will be.

Please also make use of the dynamic selections facility. Dynamic selections also directly affect response times.

Interactive List Display Functions

There is a range of interactive functions for every list that is displayed in the screen. The list can be displayed differently with some of these functions.



You can use these functions if you choose *ABAP List* in the query output selection screen.

List Overview

If your list consists of several parts, e.g. a basic list, two statistics and a ranked list, SAP Query allows you to choose one of these list types directly. To do this, proceed as follows:

1. Select *List overview*. You then see an overview of the sublists.
2. Place the cursor on the desired sublist and select *Choose*. You then return to the list display screen that contains the sublist you selected.

Selections

If you want to know what selections were entered on the selection screen, use the *Selections* function. This is particularly useful with saved lists (see [Saving and Redisplaying Lists \[Seite 113\]](#)).

Drilldown Functionality for Multiple-Line Basic Lists

With multiple line basic lists, you can sometimes choose between an expanded format and a compressed format.



A multiple line basic list is a list containing several lines with a different structure.



Suppose a basic list is defined with four lines. The first three lines contain information about a single flight and the fourth line contains information about a booking. In the expanded format of the basic list, the flight information is followed by all the bookings for this flight. The compressed format of the same list would contain the flight information, but not the bookings.

The *Expand basic list* and *Compress basic list* functions allow you to switch between a full display and a compressed display. You can completely display the information that belongs to lines of a compressed basic list using the menu path *Edit → Detailed view*.



In terms of the basic list in the above example, choosing *Detailed view* for a specific flight would produce a details list for the flight containing all the relevant bookings.

By selecting *Back*, you can return to the original list.

Compression may extend over several levels. Whether and to what extent this is possible, and which lines of a multiple line basic list are output in a compressed list, depends on the structure of the logical database to which the query refers. You cannot influence this when defining a query unless you modify the structure of individual lines.



Compression can only be used on basic lists with numerous entries. However, a multiple line definition of a basic list is still not sufficient on its own to permit

Interactive List Display Functions

compressed display, that is there are also multiple line basic lists which cannot be displayed in compressed format.

Displaying Totals in Basic Lists

If you are summing fields in a basic list, the overall total is always output. Sub-totals are also output for the different sort levels, if defined. You can suppress the individual values used to generate the totals with the *Display totals only* function. To return to the original list, select *Back*. This is also valid for the counter (see [Sorting and Sub-totals \[Seite 170\]](#)).

Compressing Statistics

By selecting *Compress statistic*, you can reduce the output of a statistic to the (overall) totals line, which is always the last line, plus any sub-total lines you may have defined. Depending on how many sort levels with sub-totals exist, you can gradually suppress the sub-total lines of the lowest sort level (i.e. the sort level with the greatest sort number) by repeatedly activating the *Compress statistic* function. Conversely, by activating the *Expand statistic* function in the same way, you can redisplay the complete statistic.



The function always manipulates the statistic, which is either indicated by the cursor or appears first on the screen.

Display as Table

The *Display as table* function allows you to display [Interactive Basic Lists \[Seite 348\]](#), statistics and ranked lists in tabular form with the help of the table view control facility. This type of display does not include totals lines, sub-totals line and lines with control level texts.



The function always manipulates the statistic, which is either indicated by the cursor or appears first on the screen.

The advantage of this display over the list is that it gives you access to a range of interactive functions. These include functions, which are automatically provided by the table view control facility (e.g. column switching, storage of settings, etc.). With other functions, you can manipulate the displayed dataset by selecting lines and columns. The functions are:

- **Convert**
This function allows you to convert currency amounts or quantity specifications to a reference currency or reference unit by marking the column containing the amounts to be converted. The currencies or units must also be in the table.
- **Sort**
Two important functions are those allowing you to sort in ascending or descending order. The sort proceeds from left to right and depends on the column(s) you select; i.e. the marked column on the extreme left of the display becomes the highest sort criterion. The sorted columns are highlighted in the display. By switching and selecting columns, you can thus sort the data any way you like.
- **Summing**
The *Sum* function allows you to calculate totals. It calculates an overall total for each numeric column and sub-totals for all selected and sorted columns. So, to obtain sub-totals for a particular sort criterion, you must first sort by selecting the relevant column (and, if necessary, other columns) and calling one of the sort functions. You then select the column again and call the *Sum* function.

All totals lines are highlighted. By using the *Display totals only* function, you can restrict the display to sub-totals and the overall total. To remove all totals lines, select *Switch off totals*.

- Find

The *Find* and *Continue search* functions allow to scan selected columns. It is not possible to search in numeric columns. Neither is a search carried out in the totals columns.

The *Find* function first requires you to enter a search criterion and then begins the search in the first visible line or in the first line in the table. If a match is found, the cursor is placed on the found term. The table is then scrolled so that the found term is displayed in the first visible line.

The *Continue search* function always searches for a previously entered search criterion. If the cursor is somewhere in the table, the search always starts from the cursor position. Otherwise, it starts from the first visible line of the table. By using the *Find* function and then repeatedly pressing *Continue search*, you can scan a table for each occurrence of a particular term.

If the term is not found or no more occurrences are found, you get an appropriate message.

- Show/hide lines and columns

The *Hide lines/columns* function allows you to hide lines and columns. You can thus restrict a display to relevant sections and make it easier to read. The sort, sum, search and print functions always refer to the visible dataset; i.e. hidden lines and columns are ignored.



You can store settings with hidden lines as a variant of the table view control.

The *Show lines/columns* function allows you to redisplay hidden lines and columns.

- Fixing columns

Two further functions allow you to make certain column positions fixed or to cancel the fixing. Although not all columns of wide lists are visible in the table view control, you can use the horizontal scroll bar to display those columns, which cannot be seen. Initially, all fields are moved when you scroll horizontally. If you then select the *Fix column* function, the selected column and all those columns to the left of it are fixed; i.e. horizontal scrolling has no effect on them. Before you use this function, all columns to be fixed must be in their original positions. To release columns from their fixed positions, select *Cancel fixing*.

- Print

The *Print* function allows you to print the table as it appears on the screen. This means that when you print out a screen list, only the visible lines and columns, as well as the sorting, totals lines, column switches and changes in column width are taken into account.

- Display list in its original form

The *Initial display* function allows you to display the table in its initial form that is exactly as when it was first called. Any subsequent steps (such as sorting, summing, hiding of lines and columns, etc.) are reversed or discarded.

Interactive List Display Functions**Display in SAP List Viewer**

An interactive function called *SAP List Viewer* is now available for all sublists in a query list ([Interactive Basic Lists \[Seite 348\]](#) , statistics, and ranked lists). This function transfers data to the SAP List Viewer from the sublist you have selected.

The *SAP List Viewer* function works in the same manner as other interactive functions that pass data. The system passes either the sublist the user has selected with the cursor, or, if this is not the case, the first sublist visible on the screen. Whenever passing the initial sublist is allowed, an additional radio button for this function exists on the selection screen, allowing you to branch directly to the SAP List Viewer.

Further information about the List Viewer can be found in the documentation under *Introduction to the R/3 System*.

Saving and Redisplaying Lists

When you execute a query, SAP Query evaluates the data from a database and outputs the results in a list. This list disappears as soon as you exit the list display. If you want to redisplay the same list later, you have to execute the query again with the same selections. And the system has to re-evaluate the database. This is unsatisfactory, especially if you are dealing with lists such as year-end closings, which do not change, but you need to display them often.

SAP Query allows you to save any lists you have generated and to redisplay them later. When you display a list, which has already been saved, no more database accesses are necessary. For this reason, the response time is considerably faster than recreating a list by executing the query again.

Saving Lists

Execute a query of your choice. For example, suppose you want to start the query G1 from the user group FB (for information about how to create this query, see the next chapter). After you have entered your selections and selected *ABAP List* for the list output, a list similar to the one below is displayed.

Display of all possible flight connections					
From city	Dest.	Flg	Nr	Depart.	Arrival
NEW YORK	SAN FRANCISCO	AA	0017	13:30:00	16:31:00
FRANKFURT	NEW YORK	AA	0026	08:30:00	09:50:00
SAN FRANCISCO	NEW YORK	AA	0064	09:00:00	17:21:00
NEW YORK	SAN FRANCISCO	DL	1699	17:15:00	20:37:00
SAN FRANCISCO	NEW YORK	DL	1984	10:00:00	18:25:00
FRANKFURT	NEW YORK	LH	0400	10:10:00	11:34:00
FRANKFURT	NEW YORK	LH	0402	13:30:00	15:05:00
FRANKFURT	SAN FRANCISCO	LH	0454	10:10:00	12:30:00
SAN FRANCISCO	FRANKFURT	LH	0455	15:00:00	10:30:00
FRANKFURT	BERLIN	LH	2402	10:30:00	11:35:00
BERLIN	FRANKFURT	LH	2407	07:10:00	08:15:00
BERLIN	FRANKFURT	LH	2415	09:25:00	10:30:00
FRANKFURT	BERLIN	LH	2436	17:30:00	18:35:00
FRANKFURT	BERLIN	LH	2462	06:30:00	07:35:00
BERLIN	FRANKFURT	LH	2463	21:25:00	22:30:00
ROM	FRANKFURT	LH	3577	07:05:00	09:05:00
NEW YORK	SAN FRANCISCO	UA	0007	14:45:00	17:55:00
FRANKFURT	SAN FRANCISCO	UA	0941	14:30:00	21:06:00
SAN FRANCISCO	FRANKFURT	UA	3504	15:00:00	10:30:00

To save this list, choose *Save*.

You then see an overview of all the lists already saved for this query:

Saving and Redisplaying Lists

List Backup for Query G1 of User Group FB	
31.08.1999 15:24:33	All flight connections
Existing saved lists	
31.08.1999 15:24:02	Flight connections of airline AA
31.08.1999 15:23:41	Flight connections of airline DL
31.08.1999 15:22:59	Flight connections of airline SQ

On this screen, you identify each list by the date and time it was created. There is also an explanatory text for each to help you locate the right list.

Your current list is the first entry. The system shows the date it was created. You have to add the explanatory text.

To save the text and the list, select *Continue*. This returns you to the display of your current list.

Redisplaying Lists

To redisplay a list, which has already been saved, select *Goto* → *Saved lists* on the initial screen

You then see an overview of the lists already saved for that query.

Saved Lists			
Query G1	User group FB		Records
<input type="checkbox"/> 31.08.1999 15:24:02	Flight connections of airline AA		1
<input type="checkbox"/> 31.08.1999 15:23:41	Flight connections of airline DL		1
<input type="checkbox"/> 31.08.1999 15:22:59	Flight connections of airline SQ		1

To choose and display a list from the overview, place the cursor on the desired list and choose *Display*. The saved list is then displayed on your screen.



If you want to know which selections were used to create the list, use the *Selections* function.



When saved lists are displayed, not all conditions may be reproduced as you remember them. Colors, for example, are not always the same. Also, symbols and icons are not displayed in saved lists. To get the original display, you must generate the list by executing the query again.

To delete any lists you no longer need, select them in the first column and use the *Delete sel. lists* function.

For each saved list, you can see how many records are occupied in background storage. Each record contains about 2900 bytes.

If you want to execute one of the interactive functions described in the section [Interactive Functions for Further List Processing \[Seite 117\]](#), the system requires certain parts of the generated query report. If any changes you make between saving the list and redisplaying it affect the structure of the sublists by altering the sequence of the fields displayed, you may not be able to execute these interactive functions. However, you can still display the list.



You cannot use interactive functions to display saved lists (see [Interactive List Display Functions \[Seite 109\]](#)) except for when using the function *Interactive list*.

Calling Other Reports

Calling Other Reports

Reports residing in different applications can call each other in the R/3 System across the Report-Report-Interface RRI. The term 'report' is used here to include all of the following: ABAP reports (including query reports), transactions, Report Writer reports, EIS drill-down reports and report portfolio reports.

The query is integrated into this interface. This means that query reports can both be called via this interface (as recipient) and can themselves call other reports (as sender). Before a query can be used as a sender or a recipient it has to be made known to the interface. You should refer to the remarks on using the Report-Report-Interface in [Assigning Title, Format, and Notes \[Seite 139\]](#).

For query lists there are two interactive functions in the *Goto* menu which allow access to the Report-Report-Interface.

- Call report

Reports are called using the interface either over the menu path *Goto → Call report* or by double-clicking (F2) on the appropriate line. Before a further report can be called successfully, the query (or query report) has to be entered in the interface as sender together with the recipient reports assigned to it. If only one report has been entered as recipient in the interface for this query, then this report will be called automatically. If more than one report has been entered, a pop-up window will appear in which one of the reports has to be selected.

The data which is to be passed to the recipient report is prepared in the *Call report* function. This data includes all the selection criteria and parameters which were entered in the selection screen of the query. Other data is determined from the position of the cursor in the query list. The cursor must always be positioned on a field. In the case of a single-line list (single-line basic list, statistics, ranking list) all the fields in the line on which the cursor is positioned are passed together with their values. If the list is a multiple line basic list then only the field on which the cursor is positioned is passed together with its field value.



Data is passed only if the corresponding field has a Dictionary reference, so that either a data element or a domain can be allocated to the data.

The data determined as above is passed across the interface as selection data to the report which was called. The data is allocated to the selection criteria of the report according to the data elements and domains. The recipient report is then executed with these selection criteria.

- Calling sequence

The function *Goto → Calling sequence* allows the user to navigate through a sequence of reports that have been called via the interface. A window appears containing a list of all reports in the calling sequence. A report can be chosen out of this list and marked as a return point.

The Report-Report-Interface also implements a quite different type of drill-down technique to that described in [Interactive List Display Functions \[Seite 109\]](#). Since this interface sees every query as a report, it is possible to combine several queries with each other quite simply via the interface.

Interactive Functions for Further List Processing

SAP Query also provides a range of interactive functions allowing you to pass lists (or, more precisely, data retrieved by a query and displayed in lists) to other software products for further processing: Further information is contained in the following sections.

[Common Principles when Passing Lists for Further Processing \[Seite 118\]](#)

[Download to File \[Seite 119\]](#)

[Table Calculation \[Seite 120\]](#)

[Graphics \[Seite 121\]](#)

[Word Processing \[Seite 123\]](#)

[ABC Analysis \[Seite 124\]](#)

[EIS \[Seite 128\]](#)

[Private File \[Seite 129\]](#)

[Additional Function Pool \[Seite 130\]](#)

[Direct Interaction \[Seite 132\]](#)

Common Principles when Passing Lists for Further Processing

The interactive functions for further processing lists are governed by certain common principles:

- These functions can handle any statistics and ranked lists, and can also be used with most basic lists as well. For more information, please refer to the section on [Interactive Basic Lists \[Seite 348\]](#).
- You can also call these interactive functions from the display with the table control. However, it is important to remember here that hidden columns are always passed on for further processing, but hidden lines are not.
 - If a list in SAP Query consists of several sublists (a basic list, statistics and ranked lists), you can pass only one of these for further processing. This is always the sublist where the cursor is when you call the function or the first sublist to be displayed on the screen.

Download to File

The *Download to file* function allows you to download the data retrieved by the query to a local file on the presentation server. The function module DOWNLOAD performs this process. When you activate the function, you see a dialog box where you can specify the file name and the data format. The system uses the contents of the input field located next to the radio button *Download to file* on the selection screen as a default for your file name.

If no file name was entered on the selection screen, the system chooses a default name for you.

You can choose the check box *With column header* in the dialog box. This will save your data with column headers if its data format so allows (DAT and DBF).



Whereas the *Download* function available in the *System* menu (choose *System* → *List* → *Save* → *Local file*) stores the list on the presentation server as an ordinary text file without structured list lines, the *Download to file* function also takes the line structure into account.

You may enter a default file name for the file created. The system uses the contents of the input field located next to the radio button *Download to file* on the query's selection screen as a default for your file name. This default value will always be as the file name whenever you call the *Download to file* function, regardless of whether the *Download to file* radio button is set on the selection screen or not.



Be aware that by entering a suggested file name, you ARE NOT suppressing the dialog that is run through when the *Download to file* function is called.

The ability to suggest names for *Download to file* files on the selection screen is especially useful when variants have been defined for your query or QuickView.

Table Calculation

Table Calculation

This function exports data via the XXL interface. If you activate it, you see a dialog box where you can decide how you want to proceed further with the data. You can

- pass the data to a table calculation program (Excel, for example) and start the program using this data or
- store the data as a file for a table calculation program or
- store the data as a SAPoffice document.

The connection to table calculation programs described here requires these programs to be installed according to the XXL interface guidelines. For further information on how to perform this installation, refer to the relevant documentation.

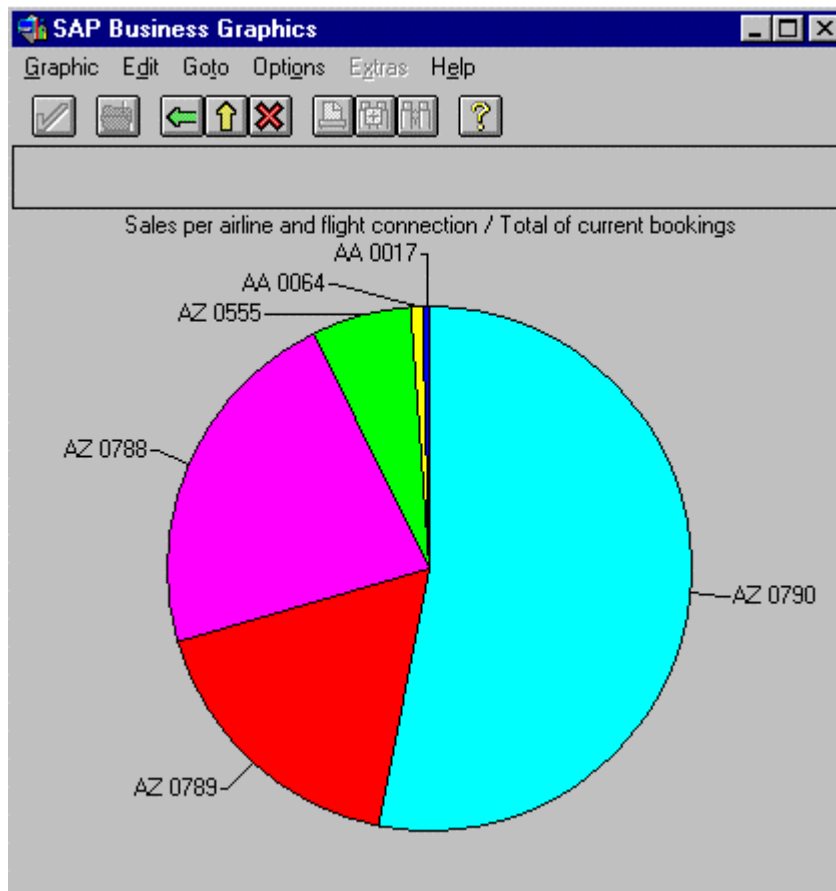
The option to start the table calculation program is not available if the start itself is not possible.



If you pass a basic list, the lines at the beginning of the control level and all sub-totals lines and overall totals lines are not taken into account. If you pass a statistic, the totals line is not taken into account.

Graphics

This function allows you to display the information from your list using SAP Business Graphics.



In contrast to the other functions described in this section, the *Graphics* function can only handle an extract from one column of your sub-list and the column must contain numeric values.

You can choose which values you want to display by placing the cursor on them when activating the function. Position the cursor on the column of the partial list whose values you want to display in graphic format. You should note here that SAP Business Graphics can handle only a limited number of values (between 6 and 32). Since the cursor position determines the first value to be processed, place the cursor directly on that value. The function then processes the value together with all the subsequent values in the same column. You can define the type of graphic in the query definition, but you can only modify the display type on the graphics screen.



If you do not position the cursor as described above when activating the function, the system itself determines a start value somewhere near the cursor position. For this reason, you should always place the cursor so that you make clear which values you want to display in graphic format.

The type of graphic can be specified in the query definition (see [Graphics \[Seite 161\]](#)). The display style can be changed in the graphics screen. In the query definition it is possible to specify that the number of values to be displayed should be decided at runtime.

Graphics

If this kind of graphic call has been defined for a partial list of a query, then when the *Graphic* function is called at runtime a window appears in which the number of values to be displayed can be entered (type of graphic, text and color display, or the number of values to be displayed). Those options chosen on the graphics screen while the query was being defined are used as defaults.



If you want display QuickView data graphically, you enter the appropriate graphical specifications at runtime.

In addition, you can also graphically display any remainder. This remainder is the sum of all values from the selected column that were not chosen for graphic display. Since only 32 values can be displayed, you may only select up to 31 individual values if you want to display your remainder as well.



If you call the *Graphics* function from the display with the table control, the handling is different from that in the list display. To pass a sequence of values for display in graphic format, you must first select the column from which the numeric values for graphical display are to be taken. You then select all the lines from which the values for graphical display are to be taken. Remember that you can only select up to 32 lines.

Word Processing

The word processing functions can be called from the list display and from the table display (display with the help of the table control). There are two different functions available:

- Form letters using MS WORD

A WORD file is created using the data from the current partial list and is linked to another WORD document. Substitution variables can be incorporated into the WORD document and these can be filled with values from the created WORD file. Special functions are made available to MS WORD via OLE2 for this purpose.

This form letter functionality can be used only if MS WORD has been installed on the PC.

- Creating a document

The current partial list can be stored on the presentation server as a file in RTF format. The use of this format allows the file to be used as a WORD document. One can specify whether the colors in the list should be taken into account. The filename (the complete path) must be specified in the dialog. If MS WORD is installed on the PC then it can be used immediately for editing the file created.



Only the form letter functionality can be called from the table display. Suppressed lines and columns will not appear in the WORD document.

ABC Analysis

ABC Analysis

The *ABC Analysis* function can be used for any interactive basic list, all statistics and any ranked list that contains at least one numeric field.

For the ABC analysis the data records of a dataset are divided into three groups (A segment, B segment, C segment). Each of the data records must include a key figure (a numeric field) and an arbitrary number of other characteristics (non-numeric fields). The dataset is initially sorted on the key figure in either descending or ascending order. The data records can then be divided into groups according to one of four strategies:

- **Key figure (percentage)**

The data records are divided into groups according to the size of the partial sum of the key figures considered as a percentage of the sum of all the key figures. For example, if the following distribution is chosen: A=50%, B=30% and C=20%, then the data records are allocated to segment A as long as the sum of the key figures does not exceed 50% of the total. The data records will then be allocated to segment B until the partial sum of the key figures reaches 80% (50+30) of the sum of all the key figures. All other data records will be allocated to segment C.
- **Key figure (absolute)**


The data records are divided into groups on the basis of the absolute values of the key figures. Two boundary values have to be specified, one to mark the boundary between the A and B segments, and one to mark the boundary between the B and C segments. These boundary values are used to determine the segment to which each data record should be allocated.
- **Characteristics (percentage)**

The data records are divided into groups so that a specific percentage of the data records are allocated to each group. For example, if the following distribution is chosen: A=50%, B=30% and C=20%, then the data records are allocated to segment A until 50% of the total number of data records have been allocated. The data records will then be allocated to segment B until the partial sum of the key figures reaches 80% (50+30) of the sum of all the key figures. All other data records will be allocated to segment C.
- **Characteristics (absolute)**

A specific number of data records are allocated to each group. For example, if the following distribution is chosen: A=10, B=20 and C=... , then the first 10 data records are allocated to segment A, the next 20 to segment B and all remaining data records are allocated to segment C.

When the *ABC Analysis* function is called the column containing the key figures must be determined first. This depends on whether the ABC analysis is called from the list display or from the table display (display using the table view control).
- **Calling from the list display**

Position the cursor on the column whose values are to be used as key figures. If you have placed the cursor on a non-numeric column then the next numeric column will be used.



The ABC analysis cannot be carried out for additional fields in statistics and ranked lists (number of records read, percentage of total, average value, position in sequence).
- **Calling from the table display**

Select the column whose values are to be used as key figures.

All non-numeric values to the left of the key figure are regarded as characteristics. This is especially important when the analysis is called from the table display, since columns can be moved around in this display and it is therefore possible to change the number of values regarded as characteristics.

When the *ABC Analysis* is first called a dialog window appears. The sort order for the key figure (i.e. ascending or descending) and the analysis strategy to be employed have to be specified in this window. This window also contains the names of the fields which are to be used for the key figure and the characteristics.

The analysis strategy is defined in two steps:

1. Select the strategy required using a selection button (see above).
2. Supply the other information required to implement the chosen strategy (percentages, boundary values or numbers of data records, as required in the strategy). When specifying percentages or absolute numbers of data records you only need to supply two values, since the third value will be calculated from the first two.

If the key figure is a currency field or quantity field, then a check is made in single line basic lists that the currencies or quantities have been specified in the same currency or unit of measurement. This check can be made only if the currency or unit of measurement is also output in the list. When different currencies or units of measurement are present, these are first converted into a reference currency or unit of measurement. This reference currency or unit of measurement can be entered in a dialog window. If errors occur in the conversion, then the ABC analysis cannot be carried out.

The result of the ABC analysis is a list. The first column of this list shows the segment (A, B or C). Then come the characteristics and finally the key figure. Next to each key figure there are two percentages: the percentage of the total sum of all key figures represented by this key figure and the cumulative total to this point. The list also includes the total sum and the intermediate sums related to the individual segments as well as the sums of the A and B segments and of the B and C segments.

ABC Analysis

Analysis strategy: Key figures (percentage)					
A segment: 50%		B segment: 30%		C segment: 20%	
ABC	Flg	Nr	Total of curr. booking DEM	%	Cumulative %
A	AZ	0790	20.032.523,16	47,0	47,0
A	AZ	0788	8.399.937,22	19,7	66,6
A			28.432.460,38	66,6	
B	AZ	0789	6.486.909,39	15,2	81,8
B			6.486.909,39	15,2	
A+B			34.919.369,77	81,8	
C	AZ	0555	2.357.373,43	5,5	87,4
C	QF	0005	988.885,74	2,3	89,7
C	UA	0941	925.427,34	2,2	91,9
C	UA	3504	560.876,22	1,3	93,2
C	LH	0402	526.259,88	1,2	94,4
C	QF	0006	428.745,50	1,0	95,4
C	SQ	0158	305.263,75	0,7	96,1
C	AA	0064	265.781,25	0,6	96,7
C	SQ	0002	237.847,49	0,6	97,3
C	DL	1699	202.427,71	0,5	97,8
C	DL	1984	200.678,57	0,5	98,2
C	SQ	0866	168.083,96	0,4	98,6
C	LH	2407	159.186,70	0,4	99,0
C	AA	0017	153.822,21	0,4	99,4
C	SQ	0988	153.782,85	0,4	99,7
C	LH	2402	112.253,25	0,3	100,0
C	LH	0400	0,00	0,0	100,0
C			7.746.695,85	18,2	
B+C			14.233.605,24	33,4	
A+B+C			42.666.065,62	100,0	

The following interactive functions can be used on the result list of the ABC analysis.

- *Hide columns*

The *Hide columns* function can be used to remove chosen characteristics columns from the display. Place the cursor on the characteristics column to be hidden and then call the function.

- *Show columns*

The *Show columns* function can be used to re-display columns which were hidden with the *Hide columns* function.

- *All characteristics*

This function is of use if a list or table contains further characteristic columns (non-numeric fields) to the right of the key figure column when the ABC analysis is called. Normally these characteristics are not included in the result list of the ABC analysis. The *Edit → All characteristics* function can be used to include these characteristics in the result list. They will, however, appear to the left of the key figure.

- *Convert*

This function allows you to convert currencies and quantities into reference currencies or units of measurement. Conversion is, however, only possible if a currency or unit of measurement exists for all currencies and quantities in the dataset. The conversion is always based on the values in the dataset, so that it is also possible to perform multiple conversions.

- *Print*

The *Print* function can be used to print the result list of the ABC analysis. The printed form of the list will correspond to the current screen display, so that hidden columns will also not appear in the printed list.

- *New Analysis*

This function can be used to start a new ABC analysis on the same dataset.

The *Back* function takes you back to the list.

EIS**EIS**

This function provides a link to the Executive Information System (EIS). It transfers the data in your query to the EIS database via an interface and you can then perform further analyses. When you activate the function, you specify various options for storing the data in a dialog box. For further information, see the EIS implementation guidelines.

Private File

If you are using the version of SAP Query supplied by SAP, you do not have this function. However, the delivery system includes an enhancement (SQUE0001) which allows all customers to add their own interactive functions. The *Private file* function appears on your screen only if you take advantage of this enhancement option. To do this, you (or your system administrator) must take the necessary steps and activate the enhancement. For further information, see [Enhancement SQUE0001: Private Files \[Seite 334\]](#).

Generally, this function works in the same way as the other standard interactive functions. It gathers the data from your query together in a table and then passes it to a function module written in the course of the enhancement. What actually happens to the data is up the customer.



The *Private file* function has largely been made obsolete with the introduction of the *Additional function pool* (see [Additional Function Pool \[Seite 130\]](#)). SAP will continue to support the *Private file* function for reasons of compatibility. It is, however, recommended that you gradually re-define those functions that you have in the *Private file* so that you can add them to the *Additional function pool*.

Be aware that additional function pool functions can only be processed online in the foreground and thus CANNOT be called during background processing. In contrast, contents of the private file CAN be called during background processing.

Additional Function Pool

All interactive functions for adjusting query lists (*Display as table*, *Download to file*, *ABC analysis*, etc.) work according to the same principles: Data from a sublist together with a description of this data is passed to another program (function module) using an interface. The number of functions that can be attached to query lists in this way is not limited. The *Additional function pool* acts as a container for these functions.

The *Additional function pool* function bundles an arbitrary number of interactive functions for single-line sublists together. Each of these functions must be implemented using a function module with a defined interface. Which and how many of the functions you want to make available to the user can be determined by using the maintenance component for the *Additional function pool*. For further information, refer to the section on [Maintaining the Additional Function Pool \[Seite 301\]](#).

The following functions are delivered in the *Additional function pool*:

1. Crystal Reports
Allows you to save a query that you have created as a file on your PC (in Dbase format). You can subsequently generate a Crystal Report from the query. The report generated from your query can be formatted in numerous ways using the large number of analysis and visualization functions that are put at your disposal. You can design the report's layout and include such things as a company logo or graphics. You then save the report and can subsequently open the template you have created each time you call a query. (Note: Field names are restricted to a maximum of 10 characters in Dbase format).
2. InfoZoom
This function allows you to connect the external PC tool InfoZoom for displaying reports and altering their layout. You must also install InfoZoom on your frontend and make sure that the file name extension '.foc' (name extension for InfoZoom files) is registered. This function formats sublist data so that InfoZoom can read it and writes the data to a temporary folder on your PC. This folder is created in the directory defined in either environment variable TMP or environment TEMP. If neither TMP nor TEMP has been defined, then the folder is created in your main directory. InfoZoom is then subsequently started with the folder that has been created.
3. Display Chart
This function allows you to create two dimensional representations of sublist data. The *Display chart* function allows you to interactively choose the fields you want to represent. In the first dimension you can include any sublist field desired, in the second dimension only numeric fields.

You can also enter as many new functions of your own to the *Additional function pool* as you like. See [Maintaining the Additional Function Pool \[Seite 301\]](#).

If the *Additional function pool* contains at least one function, the *List → Additional function pool* function (and its corresponding pushbutton) is/are activated on all single-line sublists. If the *Additional function pool* contains exactly one function, then this function is automatically executed whenever the *Additional function pool* is called. If the *Additional function pool* contains more than one function, a dialog box appears listing all of the functions at your disposal with their corresponding long texts. From this list you can both choose and execute functions.



All *Additional function pool* functions delivered by SAP can be found in the list of additional functions, but are not yet active. You must activate these functions during *Additional function pool* maintenance in order to be able to use them.

Each function has a name and a corresponding long text (description). The function module found in the list is called each time you call its corresponding function.



When executing a query in the background, you can no longer use direct passing to call these functions.

Direct Interaction

Direct Interaction

It is possible to execute queries without displaying the query list on the screen. You can instead run interactive further processing functions, display the list as a table or interactive list, or save the list automatically. The section below refers to this process as direct interaction or direct further processing of the list. When using direct interaction, you should bear in mind the following points:

- The radio buttons for the interactive functions *Display as table*, *SAP List Viewer*, *Download to file*, *Word processing*, *Table calculation*, *Graphics*, *ABC analysis*, *EIS*, *Private file*, and *Additional function pool* appear on the query selection screen only if the first sub-list is a basic list, a statistic or a ranked list. The option *Download to file* allows you to include a suggested name for your file as well.
- You cannot enter the name of an *Additional function pool* function directly; it must be inserted in the field with the help of the possible entries pushbutton on the selection screen. A dialog box appears with all the available functions. You can then choose a function from this list. This function name is then automatically transferred to the selection screen.
- The checkbox for *Save with ID* is always available. There is also a text field where you can enter a description of the list to be saved. Direct further processing of the list by saving always refers to the whole list.
- With all other functions apart from *Save*, the further processing always applies to the first sub-list. When generating graphics, the text-value pairs are determined from the first line of the sub-list, where the values used are those in the first numeric column. In the ABC analysis the values of the first numeric column are used as key figures.
- You can preset radio buttons – with the exception of the *Additional function pool* - when defining a query. In the QuickViewer, use the dropdown box available in basis mode. This means that a radio button is already selected when you execute a query. You can, however, cancel the selection.
- Direct interactions are possible only if you execute the query as if the list is displayed on the screen. If you use *Execute and print* or start the report in the background, no direct interactions are allowed. The *Save*, *EIS*, and *Private file* functions are exceptions to this rule, since you can use them when executing the query in the background.
- Using one of the functions *Download to file*, *Table calculation*, *Word processing*, *EIS*, *Additional function pool* or *Private file* to pass a list on to one of the software products mentioned above is a means of data retrieval. If you execute the query in the background when using the *EIS* function, only transaction data can be transferred.
- Direct saving of a list when executing the query in the background allows you to perform extensive analyses in the background and to display the result later online. When saving the list directly in the background, the generated list is not written to the spool file and consequently cannot be printed out.

Creating and Changing Queries

This chapter explains how to create and change queries.

There are two ways to define queries:

1. The Query Painter allows you to define lists in WYSIWYG mode (What You See Is What You Get). The Query Painter can be used to construct basic lists.

Further information can be found under [Creating Basic Lists with the Query Painter](#) [Seite 134].

2. The system takes you through several screens. Here you can define the structure of basic lists, statistics, and basic lists by making entries in the appropriate fields.

Further information can be found under [Creating Queries](#) [Seite 136].



The graphical Query Painter uses various controls. It can only be used when certain hardware and software requirements have been fulfilled. The function *Settings* → *Settings...* in query maintenance allow you to choose the query construction procedure you want. If the appropriate software requirements are fulfilled, the graphical Painter always appears as the standard setting.

Creating Basic Lists with the Query Painter

If you want to create a new query, you have to enter particular information so that the system can generate your report: The following information is required:

- InfoSet
- Title and list format
- Field selection (fields used in the list)
- List layout

To create a new query, proceed as follows:

1. Call the component *Maintain Queries*.
2. Choose the user group to which you want to assign the new query.
3. Enter the desired query name.

Query names can be up to 14 characters long and should not already exist in the user group.

4. Choose *Create*.

You then see a list of the InfoSets assigned to your current user group.

5. To select the InfoSet you require, place the cursor on the InfoSet and choose *Select*.

For further information about creating InfoSets, refer to [Creating and Changing InfoSets \[Seite 218\]](#).



If many InfoSets with multiple fields are assigned to one user group, you can use the search function to find the right InfoSet. Within the specified InfoSets it is possible to perform any number of searches. The result is a displayed list of all InfoSets and fields, which satisfy the search criterion.

You may now begin to define your report. The system first takes you to a screen for [Assigning Title, Format, and Notes \[Seite 139\]](#).

Branch to the Query Painter by choosing *Basic list*. You can select the fields you want in the Query Painter.

Or choose *Next screen* before you start the Query Painter and choose the fields you want to structure your query with. This procedure is useful if you want to define local fields.



The definition of local fields allows you to generate new information from the fields in an InfoSet, without having to include an additional field.

Consult the section [Layout Mode/The Graphical Query Painter \[Seite 367\]](#) if you want to choose your fields using the Query Painter.

If you want to create local fields and include them in your query before you call the Query Painter, consult the following sections:

[Selecting Fields for Processing \[Seite 142\]](#)

[Assigning Short Names \[Seite 144\]](#)

[Defining Local Fields \[Seite 145\]](#)

[Extending the Selection Criteria \[Seite 149\]](#)

You can find further information in the section [Layout Mode/The Graphical Query Painter \[Seite 367\]](#). You can then skip over the section on field selection and selection criteria, since you will have already done this in advance.

Creating Queries

To create a new query, proceed as follows:

6. Call the component *Maintain Queries*.
7. Choose the user group to which you want to assign the new query.
8. Enter the desired query name.

Query names can be up to 14 characters long and should not already exist in the user group.

9. Choose *Create*.

You then see a list of the InfoSets assigned to your current user group.

10. To select the InfoSet you require, place the cursor on the InfoSet and choose *Select*.

For further information about creating InfoSets, refer to [Creating and Changing InfoSets \[Seite 218\]](#).



If many InfoSets with multiple fields are assigned to one user group, you can use the search function to find the right InfoSet. Within the specified InfoSets it is possible to perform any number of searches. The result is a displayed list of all InfoSets and fields, which satisfy the search criterion.

All the examples in this chapter are based on a flight reservation system and evaluate flight bookings. You should therefore choose the InfoSet FLBU.

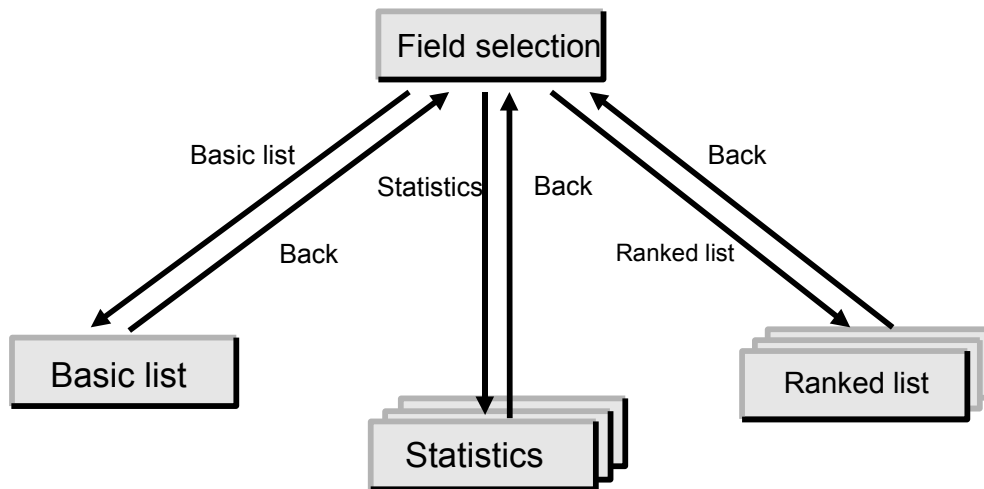
You may now begin to define your report. The system takes you through several screens. Four different screen sequences are possible:

- Field selection (including assignment of title and list format)
- Basic list (define layout of basic list)
- Statistics (define layout of statistics)
- Ranked list (define layout of ranked lists).

Within each of these screen sequences, you have the following navigation options:

- Use the *Next screen* and *Previous screen* functions.
- Use the functions in the *Goto* menu to access individual screens directly.
- Use the *Back* function to return
 - to the first screen in the sequence, if you are not already on this screen or
 - to another screen sequence, if you are on the first screen in the sequence.

If you want to switch from one screen sequence to another (for example, go from one of the basic list screens to the field selection), you can use the appropriate function in the *Goto* menu. The following screen shows the most important navigation options.



Within a query, you can define one basic list, up to 9 statistics and up to 9 ranked lists. These different list types may be combined in any permutation you like. You can also extend existing queries by adding more sub-lists. Unless you have specified otherwise, you see the basic list first, followed by the statistics and finally the ranked lists when you execute a query.

Selecting Fields

The following section describes how to make some general definitions for your query.

Assigning Title, Format, and Notes

The first attributes you must define when creating or changing a query are the title and page format. This is the first screen of the *Field selection* sequence.

Title	Display of all possible flight connections	
Notes		

List format Lines <input type="text"/> Columns <input type="text" value="83"/>	Special attributes Standard variant <input type="text"/> <input type="checkbox"/> Execute only with variant <input type="checkbox"/> Change lock
Table format Columns <input type="text" value="200"/>	Print list <input checked="" type="checkbox"/> With standard title No. of characters left margin <input type="text"/>
Output format <div style="display: flex; justify-content: space-between;"> <div> <input checked="" type="radio"/> SAP List Viewer <input type="radio"/> ABAP list <input type="radio"/> Graphic <input type="radio"/> ABC analysis <input type="radio"/> Executive Information System EIS <input type="radio"/> File store </div> <div> <input type="radio"/> Display as table <input type="radio"/> Word processing <input type="radio"/> Spreadsheet <input type="radio"/> Private file </div> </div>	

SAP Query uses the title you define in this screen later as the title of the screen or, when printing, as the first line of the heading for the lists generated by the query.

Additionally, there are three lines where you can enter notes that are used to document the query. When you request the query directory, the system displays these notes with the title.

There is a standard format for lists displayed on the screen, but you can always overwrite this. Unless you specify otherwise, the system automatically adapts the standard format to the output device used. For output to the screen, the default is a dynamic page size (i.e. no fixed number of lines) and 83 columns. In the case of output to a printer, the system proposes a standard format on the print option screen, but you can also overwrite this.

With the number of columns, you define how wide list lines can be. Regardless of this, however, the width of each sublist is no more than is actually required. The number of columns determines when a line break occurs in basic lists. With statistics and ranked lists, the number of columns must be large enough to avoid line breaks.

Assigning Title, Format, and Notes



If you define statistics or ranked lists at a later stage, the number of columns may increase automatically. In this case, you get a warning message.

The advantage of dynamic page size is that the system adapts the dimensions of your list to the size of the window. You should therefore usually use the dynamic page size. Each of the different sublists (basic list, statistics and ranked lists) always begins on a new page.

If you want to display a list as a table, you can define the size of the table view control with the table format specifications.

Special attributes allow you to define some other properties for the query or the report that it generates.

- Assigning a standard variant

You can create one standard variant per query. This variant is always used when you start the query with the help of the function *Execute*. The name of this standard variant is also the default value for the *Execute with variant* function. Any query with a standard variant is thus always executed with a variant.



Remember that you can only maintain query variants on the initial screen. For this reason, defining a standard variant is possible only when you have already defined a query to the extent that you can generate a report.

For further information about working with variants, refer to [Executing Queries Online \[Seite 103\]](#).

- Executing queries only with a variant

Besides the standard variant, there is also the attribute *Execute only with variant*. You use the standard variant only if you execute the query in the component *Maintain Queries*. However, the attribute *Execute only with variant* applies to the generated report and is therefore also effective when you start this report directly.

- Change lock

If you want to set a change lock for your query to protect it against changes by other users, you can achieve this by selecting the appropriate field. Then, only the person who set the attribute can make further changes.

By choosing *Goto → Query directory* in the component *Maintain InfoSets*, the system administrator can cancel the lock for selected queries.

- Printing lists with or without a standard title

This option allows you to decide whether you want to output a standard title, which does not appear in the screen display, for each page when printing a list. The standard title contains the date, the query title and the page numbering.



A standard title is also generated if you execute the query in the background.

If you do not want the standard title, but need the date or the page numbering, you must maintain your headers accordingly (see [Changing Headers \[Seite 158\]](#)).

- Left Margin

This option allows you to designate the size of the left margin when printing the list. This option does not affect the screen output of the list.

- Forwarding the list directly

The radio buttons on the selection screen of the generated report are pre-set as a result of the options for forwarding a list (see [Interactive Functions for Further List Processing \[Seite 117\]](#)).

The *Goto → Report assignment* function can be used to integrate the query into the Report-Report-Interface. This means that the query both can be called via this interface (as recipient) and can itself call other reports (as sender). The term 'report' is a collective term referring to ABAP reports (including queries), transactions, Report Writer reports, EIS drill-down reports and report portfolio reports.

When the *Goto → Report assignment* function is called, a window appears. The other reports that can be called from the query (if the query is the sender) are specified in this window. Since every query is also a report, this technique can be used to combine several queries with each other. This represents a new type of drill-down technique in SAP Query. The way that data is exchanged is described in [Calling Other Reports \[Seite 116\]](#).

If multiple sublists (basic lists, statistics, ranked lists) are defined for a query, you can use the function *Edit → Output sequence* to designate the order in which these sublists should be displayed. When you call this function, a dialog box appears in which all sublists that have been defined up until this point are listed in the order that they are displayed. An input field appears in front of each sublist, where a number between 1 and 90 can be entered. This sequence determines the output sequence of the sublists. Sublists with no sequence number (the standard setting) will be displayed in the order they appear in this list after those sublists with sequence numbers have already been displayed.

If two or more sublists have the same sequence number or no number at all, the order basic lists before statistics before ranked lists is used.



The function *Edit → Output sequence* is available on all maintenance screens.

To go to the next screen, select *Next screen*.

Selecting Fields for Processing

Selecting Fields for Processing

On the *Select Field* screen, you now select the fields you want to process in your basic list, statistic or ranked list, or those you want to use as additional selections.

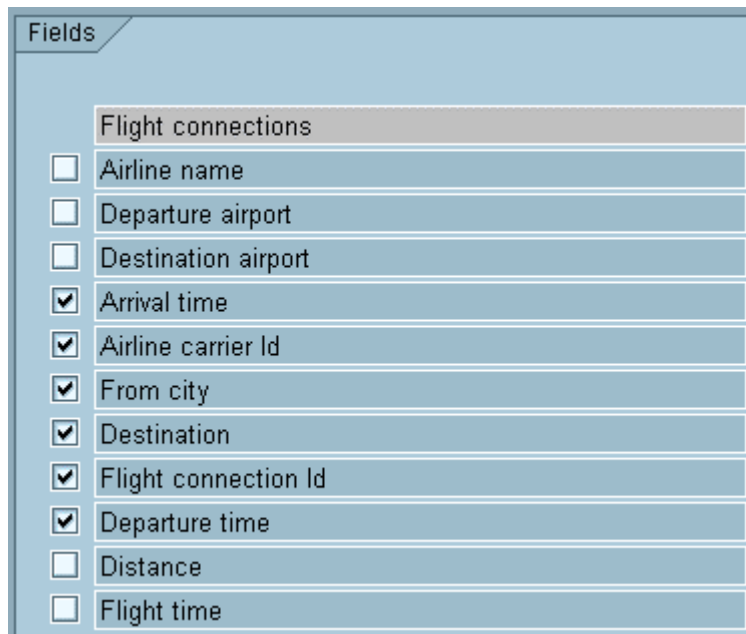
Selecting Field Groups

To facilitate pre-selection, the InfoSets are divided into field groups. If you want to generate a list of existing flight connections, you do not need any detailed information about individual flights and associated bookings. By selecting the field group *Flight connections*, you restrict the number of fields offered to those belonging to this group.

If you are editing an existing query, the *Edited* column contains all those field groups from which fields have already been selected.

Selecting Fields

On the following screen, you now select the fields you want to display in your list from the field groups selected. Select the desired fields.



Fields	
	Flight connections
<input type="checkbox"/>	Airline name
<input type="checkbox"/>	Departure airport
<input type="checkbox"/>	Destination airport
<input checked="" type="checkbox"/>	Arrival time
<input checked="" type="checkbox"/>	Airline carrier Id
<input checked="" type="checkbox"/>	From city
<input checked="" type="checkbox"/>	Destination
<input checked="" type="checkbox"/>	Flight connection Id
<input checked="" type="checkbox"/>	Departure time
<input type="checkbox"/>	Distance
<input type="checkbox"/>	Flight time

You can extend or reduce the selection of field groups and fields at any time, even if you have already defined sublists.

You can change the column headers for selected fields on this screen by placing the cursor on one of the fields and selecting *Edit ### Column headers ### Maintain* (see [Changing Headers \[Seite 158\]](#)). You then see To do this, place the cursor on the desired query and choose *Delete*. You then see a dialog box where you can change the column header or set the standard value.

By selecting *Edit → Find*, you can search for field names of the InfoSet. The result of the search is a list of all fields with names which contain the search criteria. Even here, you can select fields.

Using the function *Extras → Field documentation*, you can obtain more information about a field. To do this, place the cursor on a field name and choose this function. You then see a dialog box containing the documentation and some other technical information about the field.



This function is available on all screens where the field names are listed.

Assigning Short Names

Assigning Short Names

Each field made available through an InfoSet can have a short name. You need these short names if you want to place the value of a field in a header (see [Changing Headers \[Seite 158\]](#)) or calculate a local field (see [Defining Local Fields \[Seite 145\]](#)).



Only in these cases do you need to assign a short name.

You assign a short name to a field from an InfoSet by entering the short name in the appropriate input field. If you do not see such a field on the *Select Field* screen, choose *Edit ### Short names ### Switch on/off*.

A short name can consist of up to 10 characters. It must always begin with a letter and can contain only letters, digits and the underscore character ('_').

If you want to know whether a short name is needed, place the cursor on the field concerned or on the short name and select *Edit* → *Short names* → *References*. You then see a list of all the places in the query definition where the short name is used.



This function only returns a where-used list for the short names. It does not indicate which sublists of the query use the associated field.

Since the next section describes how to define a local field, which uses the fields *Distance* and *Flight time*, short names are assigned for both these fields.

Fields		Short name	Local
	Flight connections		
<input type="checkbox"/>	Airline name		
<input type="checkbox"/>	Departure airport		
<input type="checkbox"/>	Destination airport		
<input checked="" type="checkbox"/>	Arrival time		
<input checked="" type="checkbox"/>	Airline carrier Id		
<input checked="" type="checkbox"/>	From city		
<input checked="" type="checkbox"/>	Destination		
<input checked="" type="checkbox"/>	Flight connection Id		
<input checked="" type="checkbox"/>	Departure time		
<input checked="" type="checkbox"/>	Distance	DISTANCE	
<input checked="" type="checkbox"/>	Flight time	TIME	

Defining Local Fields

The definition of local fields allows you to generate new information from the fields in an InfoSet, without having to include an additional field.



The InfoSet FLBU contains a field for the distance and a field for the flight time. A local field is used to calculate the speed.

To define a local field, choose *Edit* → *Local field* → *Create* on the *Select Field* screen. The local field for the speed can be used as an example.

Change Query G1: Select Field

Short name: SPEED

Field description: Speed

Heading: Speed per hour

Functional group: Flight connections

Attributes:

- ☒ Same attributes as field: DISTANCE
- ☐ Text field: No. of characters
- ☐ Calculation field: Number of digits
- ☐ Date field
- ☐ Time field
- ☐ Symbol
- ☐ Icons

Calculation formula:

- ☒ Distance / (TIME / 3600)
- ☐ Condition
- ☐ Input on selection screen

☐ Mandatory

Complex calculation Fields Symbols

When making this definition, the following information is needed:

- Short name
- Field name
 - Here, you enter a text, which describes the field contents. On all subsequent screens, the local field is identified via this name.
- Header

Defining Local Fields

This is used when calculating column headers. Please note that headers are only taken into account for the output length of the field. The header can be divided over two rows.

- Field group

The local field is included in the field group.

- Attributes

When defining the technical attributes of the local field, you can use any of the following options:

- Refer to an existing field (i.e. use it as a reference field)

The advantage of this option is that the local field can accept the same values as the reference field. If the reference field is a currency amount field or a quantity field, the local field inherits the currency or unit assignment of the reference field. In our example, the field *Distance* with the short name *DISTANCE* is a quantity field, since it contains a distance specification and a measurement unit. Therefore, the local field *SPEED* is also a quantity field and has the same measurement unit as the field *DISTANCE*.

- Define the field as a text, calculation, date or time field.

With text and calculation fields, you must specify the field size (number of characters/digits and number of decimal places).

- Define the field as a symbol or icon

With these fields, the contents are interpreted as a symbol or icon and output is consequently in graphical format. You could do this if you wanted to emphasize particular values or lines in the list.

- Calculation formula

You can determine the value of a field in either of the following ways:

- By using a calculation formula.

In the simplest case, a calculation formula consists of a single formula. All formulae conform to the usual mathematical rules and contain operands and operators.

- Valid operands include the short names of fields as well as numeric constants (10 or 1.5, for example) and character strings ('ABC', for example).

- Certain special fields are also available. These are:

%NAME(the name of the user processing the query)

%DATE(the current date when the query is being processed)

%TIME (the current time when the query is being processed).

- If an operand is a text field, you can use the notation *textfield* [*n:m*] to access part of this field. Here, *n* is the position of the first character and *m* is the position of the last character.



If you were accessing the 2nd character to the 5th character of the field, the notation would be text[2:5].

- If an operand is a date or time field, you can use the following notation to access the individual components of this field:

date field [YEAR].....year

date field [MONTH].....month

date field [DAY].....day

time field [HOUR].....hours
time field [MINUTE].....minutes
time field [SECOND].....seconds

- Valid operators include the basic arithmetic operations plus DIV and MOD. DIV is the operator for whole number division and MOD is the operator for the remainder in whole number division.
- You can use parentheses in the usual way.



Please note in the example that time specifications (TIME) in calculations are always quoted in seconds.

To define the values of symbol and icon fields, you can specify the names of symbols and icons in formulae. These names begin with SYM_ or ICON_ and can be determined by pressing the appropriate pushbutton. A formula, which contains a symbol or icon, can consist only of this symbol or icon; i.e. symbols and icons cannot be linked with other operands.

You can make the calculation of the value of a field dependent on a condition. In this case, the value according to the calculation formula is only made available if the condition is satisfied. If the condition is not satisfied, the field is set to its initial value.



These types of fields are required when calculating special statistics (see [Defining Statistics \[Seite 180\]](#)).

To formulate conditions, you use Boolean formulae. These formulae consist of comparisons (e.g. AMOUNT +5 > LIMIT) which can be linked with the operators NOT, AND and OR. Parentheses may also be used in the usual way.

In more complex cases, you can make the calculation of the values of a field dependent on any number of conditions, i.e. the value is calculated differently, depending on which condition is satisfied. You can even specify a calculation formula to be used when none of the conditions is satisfied. For any of these complex cases, you use the *Complex calculation* function on the screen for maintaining local fields.

When defining the calculation formula, you can make use of different functions, which allow you to display existing short names, symbols and icons. You can also choose a symbol or an icon from the short name display and place it in the position within the formula where the cursor stands.

- By entering a value on the selection screen

In this case, the local field is treated like a parameter whose value is determined once by an entry on the screen. This is not possible for symbol and icon fields.

If you want to force input of a value on the selection screen, select the field *Obligatory*.



You can use these types of fields in conditions for calculating other local fields. This means that it is possible to influence these calculations when the query is processed. For an example of this, see below in this chapter.

When you have made all the necessary specifications for defining a local field on the screen, select *Continue* to return to the *Select field* screen. The field you have defined is now included in the specified field group and can be used on all screens, just like any other field from the InfoSet.

Defining Local Fields

An example of a complex calculation is the local field *Long text flight class*. It determines a long text for the field *Flight class* with the short name CLASS. The field is defined as a text field with 15 characters. To specify the calculation formula, you use the *Complex calculation* function to branch to the appropriate editor for such calculations.

Define Field: Complex Calculation

Condition	CLASS='C'
Formula	'Business Class'
Condition	CLASS='Y'
Formula	'Economy Class'
Condition	Class='F'
Formula	'First Class'
otherwise	

+, -, *, /, DIV, MOD, (,), [...]
 =, <, >, <>, <=, >=, AND, OR, NOT

%NAME, %DATE, %TIME
 SYM_..., ICON_...

Fields Symbols

To change or delete a local field on the *Select Field* screen, place the cursor on the local field and choose the relevant function from the *Edit* menu.



Local fields are only known within the query where they are defined.

Extending the Selection Criteria

On the *Selections* screen, you can choose the selection criteria you require when executing the query. If, for example, you select the field *Flight connection code*, you can specify in the program selections area that you only want to display flights with certain flight numbers.

If you want to make an additional selection for a field, you must select it on this screen. As a result, the selection screen is extended by one selection criterion for this field. You can also change the text for this selection criterion if you do not like the text proposed by the query.

Texts for selection criteria can only be changed after the appropriate field has been selected and *Enter* chosen.

You should use selection criteria of this type only if they cannot be implemented using dynamic selections. The advantage of dynamic selections is that the number of database accesses is reduced.

In our example (selection by *Flight connection code*), you should use dynamic selections since they are supported by the logical database.



If selection criteria are defined for InfoSet fields and local fields and these fields contain character strings (type C), then the ABAP Dictionary definition of this field determines whether case sensitivity is maintained when the selection criteria are input or if all input values are converted to capital letters (uppercase).

With additional fields and local fields whose data type definitions do not use a LIKE reference to a Dictionary field, that is with fields that have no Dictionary reference, all input values were previously capitalized by default. This meant that selections based upon case sensitivity, upon a mixture of upper and lowercase letters, could not be made in the past.

From Release 4.5A, the selection criteria for those character strings with no ABAP Dictionary reference are no longer automatically converted to capital letters. This means that when inserting selection criteria, you must now pay attention to the exact manner in which they have been written.

Defining Basic Lists

The following sections use sample lists to demonstrate the different basic list processing options that SAP Query provides.

Defining a Single-Line Basic List

A directory of flight connections is created. The query name is G1.

On the initial screen, you specify this name and select *Create*. Then, you choose the InfoSet FLBU (flight bookings) and assign a title and list format.

Since you just want to generate a list of flight connections, you only need fields from the field group *Flight connections*.

You select the fields you require (*Arrival time*, *Short description of airline*, *Point of departure*, *Destination*, *Flight connection code*, *Departure time*) by marking them.

Then, choose *Basic list*. On the next screen, you start to define the basic list.

The following sections explain how you proceed when defining a simple basic list:

[Defining the Line Structure \[Seite 152\]](#)

[List Line Output Options \[Seite 155\]](#)

[Defining Output Options for Each Field \[Seite 156\]](#)

[Changing Headers \[Seite 158\]](#)

[Graphics \[Seite 161\]](#)

Defining the Line Structure

Defining the Line Structure

On the *Basic List Line Structure* screen, you specify the lines in which you want to display and the order in which you want the fields to appear.

The following basic list contains all the fields on one line.



This basic list is also known as a single-line basic list.

☒ Basic list with box Frame width

☒ Columns separated by | ☐ Compressed display

Define basic list

Field	Line	Sequence	Sort	Total	Counter
From city	1	1			<input type="checkbox"/>
Destination	1	2			<input type="checkbox"/>
Airline carrier Id	1	3			<input type="checkbox"/>
Flight connection Id	1	4			<input type="checkbox"/>
Departure time	1	5			<input type="checkbox"/>
Arrival time	1	6			<input type="checkbox"/>

Line structure

No.+....1....+....2....+....3....+....4....+....5....+....6....+....7....+

1 |From_city_____|Destination_____|Air|Flig|Departur|Arrival_|

There are three variants of the *Basic List Line Structure* screen. The lower part of the screen for each of these variants contains different information:

1. The line structure of the basic list.

If you make changes in the upper part of the screen, these are reflected at once below (that is, after you press ENTER). To display the complete list layout, choose *Query* → *Layout display*.

2. The individual options.
3. The field display.

By using one of the functions *Settings* → *With help texts*, *Settings* → *Without help texts* or *Query* → *Line structure* and then choosing the appropriate pushbutton on the screen, you can switch between these three screens.



All the screens for defining basic lists, statistics and ranked lists have these three screen variants. Navigation between the variants is always the same.

By selecting *Settings* → *Settings...*, you can define which of the three variants you want to display first when calling a screen. Normally, the screen containing the line structure is always displayed first.

There are two additional options for improving the layout of basic lists:

- Basic list with box

Here, the basic list appears in a box. When defining the line width, you must remember that each line requires two extra characters for the box. When you leave the Frame width input field blank, the system only frames the part of the basic list actually used.

Otherwise, you may input the width of frame you desire here. If the width you enter is too big or too small, the system automatically corrects your entry to reflect the next acceptable value.

- Columns separated by |

With this option, a vertical bar is output after each field. It ensures that indented lines (for example, for sub-totals) are framed and that no lines are left open.

With multiple line basic lists, different lines usually have a different structure, so that no straight vertical lines are created when the *Column separation with |* option is selected.

You can deselect the *Columns with |* option on the *List Line Output Options* screen.

Further information can be found under [List Line Output Options \[Seite 155\]](#)

The section [Interactive List Display Functions \[Seite 109\]](#) describes how you can switch between a complete display and a compressed display with multiple line basic lists. The *Compressed display* option allows you to define whether the first list to be displayed when starting the query should be complete or compressed.



If you use this option, you specify that you want to enable a compressed display. This is then determined by SAP Query itself.

Select the options *Basic list with box* and *Columns separated by |*:

You can now execute the query by selecting *Execute*.

You then see the selection screen and get the following list after entering your selection criteria.

The *Back* function returns you to the screen where you started the query.



When you execute the query from a definition screen as explained above, the query definition is not saved. This allows you to test the list which will be generated by the specifications you have already entered. To save the query definition, choose *Save*.

Defining the Line Structure

If you now continue through the normal sequence of screens, you next see a screen where you can define output options for the lines of the basic list.

Defining Output Options for Each Line

On the *List Line Output Options* screen, you can define output options for each line.

List line output options										
Line	Color	Header line	Ref.	Slash Bef / To		Blank line Bef / Aft		Columns with	Page header	New Page
1	List line	 <input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

The column *Color* allows you to define an output format for each line. These output formats (colors) are described by terms which identify their purpose according to the SAP standard. You can only change the color for a line with the possible entries key. This shows you the color palette from which you can choose a new format.

If you select the column *Header*, the system makes column headers available for this line. These headers are displayed in a later screen. You can change the headers here too.

In order to allow for better layout in basic lists using both diagonal and vertical lines, the new options *Slash before/after* and *Columns with |* have been added.

The options *Slash before* and *Slash after* can be set for each line and determine where a diagonal line is inserted when it is displayed. This function can only be used in conjunction with the *Basic list with border (frame)* option on the *Basic list line structure* screen. (See [Defining the Line Structure \[Seite 152\]](#)) This option is activated automatically whenever the options *Slash before* and *Slash after* are selected.

The option *Columns with |* allows you to determine whether or not the various columns of a particular row should be separated by vertical lines. This option can only be used in conjunction with the *Column separation with |* option on the *Basic list line structure* screen. If this option is not activated, then all *Columns with |* options are set to inactive and cannot be reactivated. If this option is activated, then all *Columns with |* options are initially active and can be individually deactivated.

You can also specify whether you want to output blank lines before or after a line. This option and others are described in more detail later.

In our example, the output format *List line* has been chosen and *Header* selected.

If you now continue through the normal sequence of screens, you next see a screen where you can define output options for individual fields.

Defining Output Options for Each Field

Defining Output Options for Each Field

On the *Field Output Options* screen, you can define output options for individual fields. This allows you to change the layout of your list radically.

Field output options									
Field	Length	Pos	Rnd	Unit	Format	Tmpl.			
	Std/New			< * >		<>0			
From city	20 20				Line color		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Destination	20 20				Line color		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Airline carrier Id	3 3				Line color		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Flight connection Id	4 4				Line color		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Departure time	8 8				Line color		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Arrival time	8 8				Line color		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Beside each field name, SAP Query displays the standard output length of a field. If you wish to modify this, enter the new output length in the column *New*.

Increasing the output length may be particularly important if you are summing over a field and the output length required by the total is greater than that of the individual field.

In the column *Pos*, you can specify an explicit output position for a field. To determine the correct position, use the ruler provided below in the part of the screen showing the line structure.

In the column *Unit*, you can specify whether you want to output the relevant currency or quantity unit before or after currency amount fields and quantity fields, or not at all.



Currency amount fields and quantity fields are numeric fields which each have an assigned currency field or unit field. The correct interpretation of the values of one of these fields depends on the currency or unit.

The assignment of amount fields to unit fields is defined in the ABAP Dictionary and evaluated by SAP Query.

In the column *Color*, you can assign a color to each field. This may differ from the color of the line defined on the previous screen.

As with assigning a color to a line, you assign a color to a field with the possible entries key. The standard color value assigned to each field is the *Line color*. This means that the field is output with the color defined for the line.



You only have to maintain values in the column *Color* if you want to output a field with a different color from that defined for the line. If you select *Query* → *Layout display*, you can display the line structure and the color attributes.

The color *Key* has a special meaning. It ensures that any field output with this color can never be shifted horizontally in the display. With lists which are wider than the window where they are displayed on the screen, it is therefore easy to define key columns which always remain in the visible part of the screen when the list is scrolled horizontally.

Please note that, in internal processing terms, the fixed area of a list is defined for each page. Within a page, the fixed area cannot vary from line to line. SAP Query uses the key column specifications to calculate a maximum character position up to which you can go when scrolling horizontally in the visible area of the screen. With multiple line basic lists, it is possible that only part of a field belongs to the fixed area of a page.

Defining Output Options for Each Field

You can define special output templates for individual fields. This is demonstrated in a later example.

The `<>0` option allows you to suppress the output of numeric fields containing 0. With non-numeric fields (character strings), leading zeros are not displayed.

On the next screen, you can change the headers.

Changing Headers

Changing Headers

On the *Basic List Header* screen, you can define the page header and the page footer for the pages of the basic list.

Page header (headers and column headers)

Display of all possible flight connections					
From city	Dest.	Flg	Nr	Depart.	Arrival

Line structure

From_city_____	Destination_____	Air	Flig	Departur	Arrival_
----------------	------------------	-----	------	----------	----------

Page footer

--

This screen shows the general structure of the basic list. It contains the header line (page header), the line structure and the footer line (page footer). The header lines include fixed headers and column headers. Column headers are generated for lines for which you selected the column *Header* on the *List Line Output Options* screen.

Fixed Headers

Fixed header lines depend on the structure of the query. You can define any number of header lines.

On the maintenance screen, the header lines are ready for input and can accept any entries you make. The *Edit* menu contains functions which allow you to insert or delete lines. Blank header lines are not output in the list.

You can define fixed header lines and footer lines so that when you generate the query list, they can receive current values of certain fields. When defining a header line or footer line, this means that you must specify a character string of the type **&field**, where **field** is the short name of a field.

This functionality allows you (for example) to place a sort criterion in the header line.

You can use the following fields directly as variables:

%NAME	Name of the user processing the query
%DATE	Current date
%TIME	Current time
%PAGE	Current page number (6 characters)
%P	Current page number (3 characters)

They can be used in the short forms N,D,T and P. If these letters also occur as short names for query fields, the field values from the query is used as a replacement.

Column Headers

SAP Query proposes standard column headers for all lines. Here, the column header for one line of the basic list can consist of one or two lines, since the column header for a field can cover one or two lines.

If you do not want column headers for a column, place the cursor on the relevant line in the *Column header* area or in the *Line structure* area and select *Edit* → *Column header* → *Delete line*. If you require column headers for a line of your query, place the cursor on the appropriate line in the *Line structure* area and select *Edit* → *Column header* → *Insert line*. Additionally, you can also specify column headers on the *List Line Output Options* screen.

You maintain column headers field by field. If you do not want to use the header proposed by the InfoSet, you can define a query-specific header for each field.

The advantage of field by field maintenance is that column headers can always be updated automatically whenever the line structure changes.

The column header for one field can cover one or two lines.

Since the column header for each field is changed individually, the area for column headers is not ready for input on the maintenance screens. If you want to change a field header, place the cursor on the relevant field in the *Column header* area and select *Edit* → *Column header* → *Maintain*. You then see a screen where you can maintain the column header for the selected field.

Page header (headers and column headers)

Display of all possible flight connections

Field	Heading	Standard	header
Destination	Dest.	Dest.	

Standard header ☒ Standard header ☐



Even with the column header for a field, you can substitute the value of another field using **&field** in the header. However, please note that the column header may be no wider than the output length of the field. If substitution of the field value results in this length being exceeded, the end of the value will be truncated.

In queries, fields may only have one header. This header is always used if a column header is to be provided in the query for this field.

Footer Lines

The same rules apply for footer lines as for fixed header lines. These lines are ready for input and can be changed by making entries. You can insert and delete lines using the functions in the *Edit* menu. Blank lines are not output in the list.

Changing Headers

For technical reasons, it is not possible to define a different number of footer lines in the different sublists. If you have defined four footer lines for a basic list and one footer line for a statistic, the statistic is output with five footer lines. The first line of the page footer is always an underscore.

Even within a footer line, you can access the values of fields from the query.



Please note that, although the number of header lines in the page header and footer is unrestricted, the total number of these lines must be smaller than the number of lines per page. You can only check this partially in the query definition, since the number of lines per page may change at processing time (for example, when printing or when processing in the background). If this condition is violated at processing time, the program is terminated.

On the last screen in the sequence for defining a basic list, you can set the graphics parameters.

Graphics

As mentioned in [Executing Queries \[Seite 102\]](#), one-line basic lists, statistics and ranked lists can be displayed in graphical format. On the *Graphics*, you can define this graphical output depending on the values you choose to display and the graphic type you use.

By specifying the *Graphic type*, you can determine in which format the graphic is to be displayed.



Remember that some graphic types (for example pie charts) cannot display negative number values.

To generate a graphic, text/value pairs are passed to SAP Business Graphics. When calling the graphic, you choose the values by placing the cursor accordingly (see [Executing Queries \[Seite 102\]](#)). The relevant texts are generated automatically. The text for a particular value derives from the contents of the non-numeric fields in the line of the list where this value occurs.

Under *No. of displayed values*, you specify how many number values you want to display in the graphic. SAP Business Graphics can display up to 32 text/value pairs. All the specifications you make on this screen are initial values for the graphic display. You can change these values using the functions in the SAP Business Graphics menu.

If you mark the field as *Define at runtime* then you can specify the number of values to be displayed when the query is executed (type of graphic, text and color display, or the number of values to be displayed). Those options chosen on the graphics screen are used as defaults.

The option *Define at runtime* is always set to default every time a basic list, statistic or ranked list is re-defined.

Sorting

Sorting

Suppose you want to generate a list of flight connections sorted by *Point of departure*, *Destination* and *Departure time*.

To do this, you can use the existing query G1. Copy G1 and assign to it the name G2. You can now change the copy. Enter a note to document the difference to G1, for example: **Sorted by point of departure, Destination and Departure time.**

You can select more field groups and fields. The fields, that have already been selected, suffice for the directory, however. On the screen where the basic list line structure is defined, you specify the sort sequence by entering it after the corresponding fields (Point of departure: 1, Destination: 2, Departure time: 3) in the column *Sort*.

Field	Line	Sequence	Sort	Total	Counter
From city	1	1	1		<input type="checkbox"/>
Destination	1	2	2		<input type="checkbox"/>
Airline carrier Id	1	3			<input type="checkbox"/>
Flight connection Id	1	4			<input type="checkbox"/>
Departure time	1	5	3		<input type="checkbox"/>
Arrival time	1	6			<input type="checkbox"/>
Distance				<input type="checkbox"/>	<input type="checkbox"/>
Flight time				<input type="checkbox"/>	<input type="checkbox"/>
Speed				<input type="checkbox"/>	<input type="checkbox"/>

This concludes the definition of a sorted list.

You can now execute the Query. When you have entered your criteria on the selection screen, you can display the list.

Display of all possible flight connections					
From city	Dest.	Flg	Nr	Depart.	Arrival
BERLIN	FRANKFURT	LH	2407	07:10:00	08:15:00
BERLIN	FRANKFURT	LH	2415	09:25:00	10:30:00
BERLIN	FRANKFURT	LH	2463	21:25:00	22:30:00
FRANKFURT	BERLIN	LH	2462	06:30:00	07:35:00
FRANKFURT	BERLIN	LH	2402	10:30:00	11:35:00
FRANKFURT	BERLIN	LH	2436	17:30:00	18:35:00
FRANKFURT	NEW YORK	AA	0026	08:30:00	09:50:00
FRANKFURT	NEW YORK	LH	0400	10:10:00	11:34:00
FRANKFURT	NEW YORK	LH	0402	13:30:00	15:05:00
FRANKFURT	SAN FRANCISCO	LH	0454	10:10:00	12:30:00
FRANKFURT	SAN FRANCISCO	UA	0941	14:30:00	21:06:00
NEW YORK	SAN FRANCISCO	AA	0017	13:30:00	16:31:00
NEW YORK	SAN FRANCISCO	UA	0007	14:45:00	17:55:00
NEW YORK	SAN FRANCISCO	DL	1699	17:15:00	20:37:00
ROM	FRANKFURT	LH	3577	07:05:00	09:05:00
SAN FRANCISCO	FRANKFURT	LH	0455	15:00:00	10:30:00
SAN FRANCISCO	FRANKFURT	UA	3504	15:00:00	10:30:00
SAN FRANCISCO	NEW YORK	AA	0064	09:00:00	17:21:00
SAN FRANCISCO	NEW YORK	DL	1984	10:00:00	18:25:00

Output Options for Control Levels

Output Options for Control Levels

When you specify a sort sequence, the next screen in the sequence of query screens allows you to define the sort order and the output options for control levels.

If, for example, you want to sort in descending order, you must select the column *Desc* (descending).

To specify output of a blank line or a new page, you select the appropriate columns (*BlnkLn*, *NewPg.*). You can also display a complete control level, meaning all the associated lines, in a box. To do this for each of the flight connections between two places in our example, the column *Box* is selected for the *Destination* and *From city*.

Control levels	Desc	Text	Total	Cnt.	Box	BlnkLn	NewPg.
From city	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Destination	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Departure time	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

You now execute the query which generates the following list:

Flight connections						31.08.1999
From city	Dest.	Flg	Nr	Depart.	Arrival	
BERLIN	FRANKFURT	LH	2407	07:10:00	08:15:00	
BERLIN	FRANKFURT	LH	2415	09:25:00	10:30:00	
BERLIN	FRANKFURT	LH	2463	21:25:00	22:30:00	
FRANKFURT	BERLIN	LH	2462	06:30:00	07:35:00	
FRANKFURT	BERLIN	LH	2402	10:30:00	11:35:00	
FRANKFURT	BERLIN	LH	2436	17:30:00	18:35:00	
FRANKFURT	NEW YORK	AA	0026	08:30:00	09:50:00	
FRANKFURT	NEW YORK	LH	0400	10:10:00	11:34:00	
FRANKFURT	NEW YORK	LH	0402	13:30:00	15:05:00	
FRANKFURT	SAN FRANCISCO	LH	0454	10:10:00	12:30:00	
FRANKFURT	SAN FRANCISCO	UA	0941	14:30:00	21:06:00	
NEW YORK	SAN FRANCISCO	AA	0017	13:30:00	16:31:00	
NEW YORK	SAN FRANCISCO	UA	0007	14:45:00	17:55:00	
NEW YORK	SAN FRANCISCO	DL	1699	17:15:00	20:37:00	
ROM	FRANKFURT	LH	3577	07:05:00	09:05:00	
SAN FRANCISCO	FRANKFURT	LH	0455	15:00:00	10:30:00	
SAN FRANCISCO	FRANKFURT	UA	3504	15:00:00	10:30:00	
SAN FRANCISCO	NEW YORK	AA	0064	09:00:00	17:21:00	
SAN FRANCISCO	NEW YORK	DL	1984	10:00:00	18:25:00	

Output Options for Control Levels

Control levels increase the options you have for varying the layout of your list. You can, for example, output a list with the following format:

Flight connections from BERLIN 31.08.1999				
Dest.	Flg	Nr	Depart.	Arrival
Departure BERLIN				
FRANKFURT	LH	2407	07:10:00	08:15:00
FRANKFURT	LH	2415	09:25:00	10:30:00
FRANKFURT	LH	2463	21:25:00	22:30:00
Departure FRANKFURT				
BERLIN	LH	2462	06:30:00	07:35:00
BERLIN	LH	2402	10:30:00	11:35:00
BERLIN	LH	2436	17:30:00	18:35:00
NEW YORK	AA	0026	08:30:00	09:50:00
NEW YORK	LH	0400	10:10:00	11:34:00
NEW YORK	LH	0402	13:30:00	15:05:00
SAN FRANCISCO	LH	0454	10:10:00	12:30:00
SAN FRANCISCO	UA	0941	14:30:00	21:06:00
Departure NEW YORK				
SAN FRANCISCO	AA	0017	13:30:00	16:31:00

The sort criterion *From city* no longer appears on one line, but at the beginning of a control level. To generate this list, you first copy the query G2 and call it G3. You can then change the query G3. The field *From city* is not assigned to a line, but the sort is performed by this field.

<input checked="" type="checkbox"/> Basic list with box	Frame width	<input type="text" value=""/>
<input checked="" type="checkbox"/> Columns separated by	<input type="checkbox"/> Compressed display	
Define basic list		
Field	Line	Sequence
From city		1
Destination	1	2
Airline carrier Id	1	3
Flight connection Id	1	4
Departure time	1	5
Arrival time	1	6

If you select the column *Text* for the appropriate level on the *Control Levels* screen, the (control level) text is displayed at the beginning of the control level.

Output Options for Control Levels

Control levels	Desc	Text	Total	Cnt.	Box	BlnkLn	NewPg.
From city	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Destination	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Departure time	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

In this case, you generate the list shown above.

Another useful variant of this query would be one which has the sort criterion *From city* in the header. To do this, you can copy the query G3 and call it G4. To achieve this, you select the column *NewPg.* for the control level *From city* so that, when a control break occurs, (that is, the sort criterion changes), a new page is started.

Control levels	Desc	Text	Total	Cnt.	Box	BlnkLn	NewPg.
From city	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Destination	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Departure time	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

You must also assign a short name (for example, CITY) to the field *From city* (see [Assigning short names \[Seite 144\]](#)). This short name is then used as a replacement variable (&CITY) in the header (see [Changing Headers \[Seite 158\]](#)). This displays the following list.

Output Options for Control Levels

Flight connections from BERLIN 31.08.1999				
Dest.	Flg	Nr	Depart.	Arrival
FRANKFURT	LH	2407	07:10:00	08:15:00
FRANKFURT	LH	2415	09:25:00	10:30:00
FRANKFURT	LH	2463	21:25:00	22:30:00

Flight connections from FRANKFURT 31.08.1999				
Dest.	Flg	Nr	Depart.	Arrival
BERLIN	LH	2462	06:30:00	07:35:00
BERLIN	LH	2402	10:30:00	11:35:00
BERLIN	LH	2436	17:30:00	18:35:00
NEW YORK	AA	0026	08:30:00	09:50:00
NEW YORK	LH	0400	10:10:00	11:34:00
NEW YORK	LH	0402	13:30:00	15:05:00
SAN FRANCISCO	LH	0454	10:10:00	12:30:00
SAN FRANCISCO	UA	0941	14:30:00	21:06:00

Flight connections from NEW YORK 31.08.1999				
Dest.	Flg	Nr	Depart.	Arrival
SAN FRANCISCO	AA	0017	13:30:00	16:31:00

The advantage of this variant is that the sort criterion *From city* remains visible when you scroll in the generated list.

Defining Multiple Line Basic Lists

Defining Multiple Line Basic Lists

Suppose you want to generate a list showing, for each flight connection, the actual flights together with the date, price and information about occupied and free seats.

To do this, you can copy the query G2 and call it G5. Then, change the title of G5 and enter a new note.

On the *Select Field Group* screen, which is displayed next, choose the field group *Flights*. This is followed by the *Select Field* screen where you select the field that contains the required flight information. You can see these fields in the screen shown below.

Now, determine the output lines. To do this, branch to the screen for defining the basic list.

☒ Basic list with box Frame width 83

☐ Columns separated by | ☐ Compressed display

Define basic list

Field	Line	Sequence	Sort	Total	Counter
City of departure	1	1	1		<input type="checkbox"/>
Arrival city	1	2	2		<input type="checkbox"/>
Airline carrier ID	1	3			<input type="checkbox"/>
Flight connection Id	1	4			<input type="checkbox"/>
Departure time	1	5	3		<input type="checkbox"/>
Arrival time	1	6			<input type="checkbox"/>
Flight date	2	1			<input type="checkbox"/>
Airfare	2	2		<input type="checkbox"/>	<input type="checkbox"/>
Maximum capacity	2	3		<input type="checkbox"/>	<input type="checkbox"/>
Occupied seats	2	4		<input type="checkbox"/>	<input type="checkbox"/>



In this basic list, two different lines are defined. This means that the list is a two-line basic list.

With a two-line basic list, you should cancel the *Columns separated by |* option. You also want to include every individual flight connection. For this, select the column *Frame* in the following screen for the group level *Departure time*.

You can now execute your query. When you have entered your selections on the selection screen, you generate the following list:

Depart. city Flgt date	Arrival city FlgtPrice	ID No. Capacity	Depart Occupied	Arrive
FRANKFURT 28.02.1995	NEW YORK 849,00 DEM	SQ 0026 380	08:30:00 2	09:50:00
FRANKFURT 28.02.1995	NEW YORK 899,00 DEM	LH 0400 350	10:10:00 3	11:35:00
FRANKFURT	NEW YORK	LH 0402	13:30:00	15:05:00
FRANKFURT 17.11.1995	SAN FRANCISCO 1.499,00 DEM	LH 0454 350	10:10:00 2	12:30:00

Defining Multiple Line Basic Lists

This list is still insufficient. Because you only want to output a flight connection if flights exist for the selected period.

The *Back* function returns you to the query definition. Then, following the normal sequence of query screens, you go to the *List Line Output Options* screen.

In the column *Ref.*, you can specify in connection with which line you want to output the data. Since the flight connections (line 1) are only meant to be displayed in connection with the flights (line 2), you must enter a **2** for line 1 in the column *Ref.*

List line output options										
Line	Color	Header	Ref.	Slash		Blank line		Columns	Page	New
		line		Bef	To	Bef	Aft	with	header	Page
1	List line	<input checked="" type="checkbox"/>	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	List line	<input checked="" type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

With these options, you can generate a list containing only those flight connections for which flights exist in the selected period.



To output the price of a flight in the relevant currency, you must set the appropriate radio button for the field *Price* in the column *Unit* on the *Field Output Options* screen.

On the *List Line Output Options* screen, there are further list layout options, which have not so far been discussed.

For each line, you can specify whether you want to output blank lines before and after outputting this line. To do this, you enter the number of blank lines in the relevant columns.

If you select the column *New page* for a line, SAP Query starts a new page before outputting this line. In this way, you can ensure in the above example that information about a flight connection always begins on a new page. However, when displaying lists with a dynamic page size on the screen, starting a new page can be more of a hindrance (for more about "page size", please refer to [Assigning Title, Format, and Notes \[Seite 139\]](#)).

The *Page header* option is only of significance if you are defining your query with a fixed page size or want to print your query lists. This option allows you to repeat certain lines in every page header. In the above example (query G5), each flight connection (line 1) is followed by several flights (line 2). This means that if you are working with a fixed page size (which is always the case when printing), one page may contain only lines with flights. However, if you select the column *Page header* for line 1 (flight connection information), the line containing the flight connection will be output on each new page. Therefore, it is always easy to see to which flight connection the flights on each page belong.

Sorting and Subtotals

Sorting and Subtotals

In this example, suppose you want to generate a list which contains booking information for each flight. You want to display a subtotal giving the amounts paid per booking in each case and an overall total at the end of the list.

In addition, you want to determine how many bookings per flight and the total number of bookings made.

You can use G5 as the basis for this: Copy it and call it G6. Then, change the title and note.

You must now extend the field selection by a few fields from the InfoSet *Bookings*.

On the *Basic List Line Structure* screen, you extend the definition of the basic list. You also specify which field you want to sum. Since you want to sum the *Booking price in foreign currency* field, you select the column *Total* for this field.

In order to count the number of bookings, select the checkbox for this entry in the *Counter* column.

Field	Line	Sequence	Sort	Total	Counter
From city	1	1	1		<input type="checkbox"/>
Destination	1	2	2		<input type="checkbox"/>
Airline carrier Id	1	3			<input type="checkbox"/>
Flight connection Id	1	4			<input type="checkbox"/>
Departure time	1	5	3		<input type="checkbox"/>
Arrival time	1	6			<input type="checkbox"/>
Flight date	2	1	4		<input type="checkbox"/>
Ticket price	2	2		<input type="checkbox"/>	<input type="checkbox"/>
Occupied seats	2	3		<input type="checkbox"/>	<input type="checkbox"/>
Free seats	2	4		<input type="checkbox"/>	<input type="checkbox"/>
Booking number	3	1			<input type="checkbox"/>
Customer name	3	2			<input type="checkbox"/>
Flight class	3	3			<input type="checkbox"/>
Smoker	3	4			<input type="checkbox"/>
Booking price in foreign currency	3	5		<input checked="" type="checkbox"/>	<input type="checkbox"/>

If you continue with the normal sequence of screens, you now branch to the screen where you can specify output options for each control level. Since you want to display a sub-total and the number of bookings for each flight, you select the columns *Total* and *Counter* for the field *Flight date*.



On this screen, you cannot determine the field you want to sum or count, but only at which level you want to display a sub-total or number of bookings for the fields you

Sorting and Subtotals

have defined on the *Basic List Line Structure* screen for summing or counting respectively.

In addition, you have chosen the *Box* option for the field *Flight date*.

Control levels		Desc	Text	Total	Cnt.	Box	BlnkLn	NewPg.
From city		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Destination		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Departure time		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Flight date		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

On the *Field Output Options* screen, you also specify that you want to display the price per booking, including the currency.

Field output options		Length	Pos	Rnd	Unit	Format	Templ.
Field	Std/New				< * >		<=>0
From city	20 20					Line color	<input type="checkbox"/> <input type="checkbox"/>
Destination	20 20					Line color	<input type="checkbox"/> <input type="checkbox"/>
Airline carrier Id	3 3					Line color	<input type="checkbox"/> <input type="checkbox"/>
Flight connection Id	4 4					Line color	<input type="checkbox"/> <input type="checkbox"/>
Departure time	8 8					Line color	<input type="checkbox"/> <input type="checkbox"/>
Arrival time	8 8					Line color	<input type="checkbox"/> <input type="checkbox"/>
Flight date	10 10					Line color	<input type="checkbox"/> <input type="checkbox"/>
Ticket price	20 20				<input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>	Line color	<input type="checkbox"/> <input type="checkbox"/>
Occupied seats	10 10					Line color	<input type="checkbox"/> <input type="checkbox"/>
Free seats	10 10					Line color	<input type="checkbox"/> <input type="checkbox"/>
Booking number	8 8					Line color	<input type="checkbox"/> <input type="checkbox"/>
Customer name	25 25					Line color	<input type="checkbox"/> <input type="checkbox"/>
Flight class	1 1					Line color	<input type="checkbox"/> <input type="checkbox"/>
Smoker	1 1					Line color	<input type="checkbox"/> <input type="checkbox"/>
Booking price in foreign currency	20 20				<input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>	Line color	<input type="checkbox"/> <input type="checkbox"/>

To execute the query, select *Execute*. You then see a list with an overall total and number of bookings at the end:

Sorting and Subtotals

Bookings					01.09.1999	
From city	Dest.	Air Nr	Depart.	Arrival		
F1. date	Ticket price	Occupied	Free			
Booking	Customer name	C S	Amount (for.currenc)			
00000034	Ballmann, G.	F X	1.485,56	USD		
00000035	Radetzky, P.	Y	2.469,13	DEM		
00000036	Zielcke, E.	Y X	1.455,24	USD		
00000037	SAP France SA	C	1.349,13	USD		
00000038	SAP Canada	C	2.077.771	ITL		
00000039	Brucher, K.	F	1.485,56	USD		
00000040	Brucher, F.	Y	1.424,93	USD		
00000041	Francke, E.	Y X	2.320,98	DEM		
00000042	Dampf, Hans	F	1.515,88	USD		
00000043	Becker, Boris	Y X	1.394,61	USD		
Total F1-Date 10.05.1999			16.197,47	DEM	*	
			15.791.045	ITL		
			41.838,26	USD		
Number F1-Date			43		*	
00000001	Bond, James	Y	2.248.879	ITL		
00000002	Zielcke, E.	F	1.455,24	USD		
00000003	Tillinger, Hans	Y	1.485,56	USD		
00000004	SAP Asia	C X	2.419,74	DEM		
00000005	Sterk, O.	Y X	1.470,40	USD		
00000006	Ruthenberg, K.	F X	1.455,24	USD		
00000007	SAP AG	C	2.102.218	ITL		
00000008	SAP Australia Pty. Ltd	C	1.409,77	USD		
00000009	Meyer, K.	Y	1.485,56	USD		
Total F1-Date 11.05.1999			2.419,74	DEM	*	
			4.351.097	ITL		
			8.761,77	USD		
Number F1-Date			9		*	
Overall total			35.259,11	DEM	**	
			36.422.063	ITL		
			91.862,27	USD		
Total number			95		**	

If you are totaling a particular field, the total is output in the same column as the field, that is with the same output length. For this reason, the output length may be too short for the total and cause overflow. ABAP identifies these overflows by placing an asterisk in front of the affected value when it is output.

To avoid overflows like this when outputting totals, you can simply increase the output length of the field. This is possible on the screen *Output Options Field*.

In the above example, summing is performed on a field which contains currency amounts. Here, a currency distribution is generated automatically, that is, the individual currency amounts are summarized depending on their currencies.

Currency-specific summation also applies when you specify the output of the currency amount field without a unit. Then, several currencies (without a currency) appear in the totals lines. For this reason, you should always output currency amounts with a currency.

The same applies to quantity fields. If you specify summation for a quantity field, this is performed according to the units involved and results in a distribution in the totals lines.

Finally, the options *Total* and *Counter* should be compared one more time.

The *Total* option causes the grand total of a specified numeric field to be calculated. This means that each time the field is read in a dataset, the field's value is added to the sum total. The grand total is displayed at the end of the basic list. With control levels, you can also display subtotals. A subtotal contains all values of a field that are assigned to a particular control level, that is a particular sort string.

The *Counter* option causes the counter number for a particular field to be increased by 1 every time that this field is found within the data set currently being read. The resulting counter total is displayed in much the same manner as a grand total at the end of the basic list. Just as with sum totals, counter subtotals can be displayed for individual control levels. These counter subtotals show how many values have been assigned to a particular control level.



If the both the *Total* and *Counter* checkboxes have been selected for a numeric field, the counter number is always equal to the number of addends for that field. The *Counter* option can also be selected for non-numerical fields.

Changing Control Level Texts

Changing Control Level Texts

The previous examples demonstrated how you can design the layout of your list if you select the columns *Text*, *Total* and *Counter* when you are on the *Control levels* screen.

- **Text**
If you select the column *Text*, a (control level) text containing the field contents is output at the start of the control level that is whenever the contents of the affected field change.
- **Total**
If you select the column *Total*, the sub-total for all summed fields is output at the end of the control level. Here again, a (control level) text is also displayed. This describes the control level and contains the contents of the control level field.
- **Counting**
If you select the column *Counter*, this displays the number the of times the selected fields appeared in that particular dataset at the end of the control level. Here a text is displayed that describes the control level and contains the contents of the control level field.

In the examples so far, always those texts available as standard for the query have been used as control level and sub-total texts. You can, however, change these texts. When you leave the *Control levels* screen and continue with the sequence of query screens, you reach a series of screens where you can define these texts for each control level. Below is the first screen in the sequence for the query G6:

On this screen, there is a sub-total text and a counter text for the control level *Flight date*, because you selected the columns *Total* and *Counter* for this control level on the *Control levels* screen. If you had also selected the column *Text*, the control level text would also be displayed on the screen. Apart from this, you see the standard texts provided by SAP Query.

You can now change the control level texts as you like. The templates in the texts (with "<" and ">" as delimiters) are placeholders for the contents of the control level field, that is, in the list, the template is replaced by the field contents. The characters "<" and ">" are not part of the template and are ignored when outputting the field. Each underscore in the template represents a character of the field contents. If you insert other characters in the template or change the number of underscores, the field contents are re-formatted accordingly.

If you have already changed the texts and then decide you want to use the standard texts after all, you can reinstate them with the *Standard header* function.

Totals Lists

Totals Lists

The sections above described how to define lists which contain control totals. However, it is also possible to generate lists which output only control totals and overall totals.



You can always display control totals and overall totals in a secondary list by using the *Display totals only* function in the list display.

Suppose you want the query G7 to generate a list which contains the revenues, i.e. the total bookings, for each airline carrier and flight connection.

You first define the sort sequence and the fields to be summed:

<input checked="" type="checkbox"/> Basic list with box	Frame width <input type="text"/>
<input checked="" type="checkbox"/> Columns separated by	<input type="checkbox"/> Compressed display
Define basic list	
Field	Line Sequence Sort Total Counter
Airline carrier Id	<input type="text"/> <input type="text"/> 1 <input type="checkbox"/>
Flight connection Id	<input type="text"/> <input type="text"/> 2 <input type="checkbox"/>
Total of current bookings	<input type="text"/> <input type="text"/> <input checked="" type="checkbox"/> <input type="checkbox"/>

On the *Control levels* screen, you specify that you want to output sub-totals for each airline carrier and each flight connection. The column *Total* must be selected for this control level on the screen. Furthermore, the column *Frame* is selected for the *Airline Carrier ID*.

This definition generates the following list:

	Total of curr. booking			
Total Flight connection Id 0017	369.450,12	USD	*	
Total Flight connection Id 0026	368.779,32	USD	*	
Total Flight connection Id 0064	276.299,33	USD	*	
Total Airline carrier Id AA	1.014.528,77	USD	**	
Total Flight connection Id 1699	369.450,12	USD	*	
Total Flight connection Id 1984	307.875,10	USD	*	
Total Airline carrier Id DL	677.325,22	USD	**	
Total Flight connection Id 0400	401.182,72	DEM	*	
Total Flight connection Id 0402	650.712,05	DEM	*	
Total Flight connection Id 0454	601.774,08	DEM	*	
Total Flight connection Id 0455	702.069,76	DEM	*	
Total Flight connection Id 2402	802.365,44	DEM	*	
Total Flight connection Id 2407	450.960,21	DEM	*	
Total Airline carrier Id LH	3.609.064,26	DEM	**	
Overall total	3.609.064,26	DEM	***	
	1.691.853,99	USD		

Totals lists also generate an automatic currency distribution, if the summed field is a currency amount field. In contrast to single item output, however, the currency always appears after the totals. Logically, the same applies to quantity fields.

Just as in the example above, you can define a list where only the results of counter operations are displayed. You can combine both types of lists as well.

Output Positions of Fields

Output Positions of Fields

Until now, we have only specified the output line and the sequence of fields when determining the output position of fields. It is, however, also possible to position fields in a screen. You can add texts to your lists too.

For example, you can format the list of flight connections G1 as follows:

Flight connections				01.09.1999
Flight AA	0017	NEW YORK	SAN FRANCISCO	
		Depart.	13:30	
		Arrival	16:31	
Flight AA	0026	FRANKFURT	NEW YORK	
		Depart.	08:30	
		Arrival	09:50	
Flight AA	0064	SAN FRANCISCO	NEW YORK	
		Depart.	09:00	
		Arrival	17:21	
Flight DL	1699	NEW YORK	SAN FRANCISCO	
		Depart.	17:15	
		Arrival	20:37	
Flight DL	1984	SAN FRANCISCO	NEW YORK	
		Depart.	10:00	
		Arrival	18:25	

After copying the query G1 to G8, you can first change the output lines and the sequence of fields:

☒ Basic list with box
 ☐ Columns separated by |
 ☐ Compressed display

Define basic list

Field	Line	Sequence	Sort	Total	Counter
Airline carrier Id	1	1	1		<input type="checkbox"/>
Flight connection Id	1	2	2		<input type="checkbox"/>
From city	1	3			<input type="checkbox"/>
Destination	1	4			<input type="checkbox"/>
Departure time	2	1			<input type="checkbox"/>
Arrival time	3	1			<input type="checkbox"/>

You then sort by the individual flight connections and specify on the *Control levels* screen that you want each control level to be displayed in a box. No column headers are necessary in the new list. We can therefore delete the option *Header* in the *Output Options List Line* screen.

If you now proceed to the next screen in the sequence, you branch to the *Field Output Options* screen.

Output Positions of Fields

You specify a fixed output position for the fields *From city*, *Departure time* and *Arrival time*. This means they are always output from this position. The fields *Short name*, *Departure time* and *Arrival time* are output specially formatted or with text. The *Template* option is therefore selected.

Field output options						
Field	Length	Pos	Rnd	Unit	Format	Tmpl.
	Std/New					<>[]
Airline carrier Id	3 10				Line color	<input checked="" type="checkbox"/> <input type="checkbox"/>
Flight connection Id	4 4				Line color	<input type="checkbox"/> <input type="checkbox"/>
From city	20 20	20			Line color	<input type="checkbox"/> <input type="checkbox"/>
Destination	20 20				Line color	<input type="checkbox"/> <input type="checkbox"/>
Departure time	8 15	20			Line color	<input checked="" type="checkbox"/> <input type="checkbox"/>
Arrival time	8 15	20			Line color	<input checked="" type="checkbox"/> <input type="checkbox"/>

The *Tmpl.* option allows you to include formatting characters in a field. You can make your specifications for the template on the next screen. For example, you can insert a text or enter separators.

Field templates	
Fields	
Airline carrier Id	<Flight ____>
Departure time	<Depart. __:__>
Arrival time	<Arrival __:__>

The system replaces each underscore by the contents of the relevant field.

The output length of the field is increased by the number of formatting characters. When you return to the *Field Output Options* screen after entering a template, you see the new output length in the column *New*.



When you change the output length of a field, whether directly by defining a length on the *Field Output Options* screen or indirectly by defining a template, ensure that the line length defined on the *Title, Format* screen is not exceeded. If a field is too long for the line, it is displayed on a new line. When calculating the line length, do not forget the blank characters between the fields assigned by the program.

The query defined in this way generates the above list.

Defining Statistics

SAP Query also allows you to define statistics. In contrast to basic lists, the data in statistics and ranked lists is always output in compressed form. For further information, refer to the following sections.

Generating Statistics

Below is a statistic showing the sales figures for each airline carrier.

Sales per airline				
Flg	Total of curr. booking DEM	Total Number	Proportion in %	Average Value
AA	1.652.505,34	24	26,0 %	68.854,39
DL	1.103.254,57	16	17,3 %	68.953,41
LH	3.609.064,26	48	56,7 %	75.188,84
	6.364.824,17	88	100,0 %	72.327,55

Conversions in statistics/ranked lists			
From -> To	Exchange rate/conversion factor	Date	
USD -> DEM	P1,62884 1:1	01.09.1999	

For this purpose, you can create a query S1 for the InfoSet FLBU (flight bookings). After assigning a title, format and note, you then choose the field groups *Flight connections* and *Flights*.

The only fields you need are *Airline carrier ID* and die *Total of current bookings*. You select these on the following screen. Then, you call the *Statistics* function to branch to the definition of statistics.

Each statistic must have its own title, since there can be several statistics.

You can specify the sequence in which you want to output the fields and whether they should be in ascending or descending order.

Totals are always calculated for numeric fields. For this reason, you can also determine average values, the number of selected records (summands) and percentages.

Title		Sales per airline											
Define statistic													
Field	No	Srt	De	Su	NS	No	Av	%	Len	Rnd	Unit	Text	
Airline carrier Id	1		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>							<input checked="" type="checkbox"/>	
Total of current bookings	2		<input type="checkbox"/>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			DEM	<input checked="" type="checkbox"/>	

To calculate average values and percentages, you must sum over currency fields.

You must also convert the amounts to one currency - the reference currency. There are two reasons for this. Firstly, the individual currency amounts to be summed may be in several different currencies. Secondly, a statistic is only meaningful when the values in a column can be compared. The same applies when using quantity fields in statistics.

Generating Statistics

The reference currency for currency amount fields and the reference unit for quantity fields are defined for each currency amount field or quantity field. On the screen for defining statistics or ranked lists, you can define a reference currency or reference unit for each individual field.

You define the conversion date for currency amounts yourself when you execute a query by using the parameter *Date of currency conversion* on the selection screen. You may also enter an *Exchange rate type* here. All conversions are then undertaken according to this exchange rate type. If multiple lines containing currency values exist, all conversions are made according to the exchange rate type.

In addition, you can also enter a reference currency on the selection screen. Whenever multiple columns with currency values exist, all conversions are made on the basis of the reference currency. If you do not enter a reference currency, conversions are made using the reference currency entered during query definition.

You can define up to 9 statistics for each query. The *Next statistic* allow you to proceed to the statistics screen where you can define another statistic.

The annual sales in DM were output in the first statistic. We can change this by entering a number in the column *Rnd*. If you enter a **3**, the amounts are converted to amounts in thousands.

When you execute the query, the statistics you have defined are displayed one after the other.

After the statistics, you get an overview stating which currency conversions were performed, as well as the date and exchange rate on which the conversions were based.

The overview also provides information about quantity conversions and the conversion factor.

You select these on the next screen and then use the *Statistics* function to branch to definition of a statistic. They are determined by SAP Query itself. The color attribute *Key* is automatically assigned to all non-numeric fields. When you scroll horizontally, these fields always remain in the visible part of the screen.

When defining statistics, you can change the output length of fields. This may be necessary, for example, when overflows occur as a result of summing numeric fields. To do this, specify the desired output length in the column *Len* on the *Statistic Structure* screen.

Sorting Statistics

You can sort statistics both independently of each other and independently of a basic list defined in the same query. You can also generate sub-totals for sort criteria.

The following example defines a statistic showing sales figures for each airline carrier and flight connection. It outputs a sub-total for each airline carrier.

Title: Sales per airline and flight connection

Define statistic

Field	No	Srt	De	Su	NS	No	Av	%	Len	Rnd	Unit	Text
Airline carrier Id	1	1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>							<input checked="" type="checkbox"/>
Flight connection Id	2	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>							<input checked="" type="checkbox"/>
Total of current bookings	3		<input type="checkbox"/>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			DEM	

This generates the following list:

Sales per airline and flight connection					
Flg	Nr	Total of curr. booking DEM	Total Number	Proportion in %	Average Value
AA	0017	601.775,22	8	9,5 %	75.221,90
AA	0026	600.682,64	8	9,4 %	75.085,33
AA	0064	450.047,48	8	7,1 %	56.255,94
AA	*	1.652.505,34	24	26,0 %	68.854,39
DL	1699	601.775,22	8	9,5 %	75.221,90
DL	1984	501.479,35	8	7,9 %	62.684,92
DL	*	1.103.254,57	16	17,3 %	68.953,41
LH	0400	401.182,72	8	6,3 %	50.147,84
LH	0402	650.712,05	8	10,2 %	81.339,01
LH	0454	601.774,08	8	9,5 %	75.221,76
LH	0455	702.069,76	8	11,0 %	87.758,72
LH	2402	802.365,44	8	12,6 %	100.295,68
LH	2407	450.960,21	8	7,1 %	56.370,03
LH	*	3.609.064,26	48	56,7 %	75.188,84

As already shown, apart from the overall total at the end of the statistic, you can output sub-totals lines by selecting the field *Su* on the *Statistic Structure* screen. When switching a sort string a sub-total line is created for this sort string respectively. The following three conditions must, however, be met:

- Sub-totals lines are only possible for fields by which you are sorting.
- the fields in the statistics must likewise be fields with sorting. the sort sequence must be ascending
- None of the subsequent fields in the statistic may have a smaller sort number.

Sorting Statistics



You can also set the option *New page* with sort fields. A page break is then inserted after each change of sort field. The same restrictions apply to the *New page* option that apply to the option *Subtotal*.

The next example demonstrates how you can use local fields with calculations which depend on conditions in a sorted statistic. The statistic contains quarterly sales figures of individual airline carriers for a year which you specify on the selection screen.

Quarterly sales 1999				
ID	1st quarter 1999 DEM	2nd quarter 1999 DEM	3rd quarter 1999 DEM	4th quarter 1999 DEM
AA	95.410,25	167.227,05	204.368,63	578.932,00
DL	13.890,70	290.420,43	258.682,05	420.296,74
LH	155.837,18	864.842,42	1.069.144,76	2.736.026,06
UA	125.642,60	588.623,14	485.843,76	2.218.718,68
	390.780,73	1.911.113,04	2.018.039,20	5.953.973,48

Conversions in statistics/ranked lists			
From -> To	Exchange rate/conversion factor		Date
USD -> DEM	1,81240	1:1	14.09.1999

You first need to define some local fields for this statistic.

Sorting Statistics

<input checked="" type="checkbox"/>	Airline carrier Id		
<input type="checkbox"/>	From city		
<input type="checkbox"/>	Destination		
<input type="checkbox"/>	Flight connection Id		
<input type="checkbox"/>	Departure time		
<input type="checkbox"/>	Distance		
<input type="checkbox"/>	Flight time		
	Flights		
<input type="checkbox"/>	Price in local currency		
<input type="checkbox"/>	Free seats		
<input type="checkbox"/>	Flight date	FD	
<input type="checkbox"/>	Total of current bookings	PAYMENT	
<input type="checkbox"/>	Plane type		
<input type="checkbox"/>	Ticket price		
<input type="checkbox"/>	Maximum capacity		
<input type="checkbox"/>	Occupied seats		
<input type="checkbox"/>	Year	Y	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	Bookings for the 1st quarter	Q1	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	Bookings for the 2nd quarter	Q2	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	Bookings for the 3rd quarter	Q3	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	Bookings for the 4th quarter	Q4	<input checked="" type="checkbox"/>

The field **Year** is a local field which you specify on the selection screen. Its value determines the year for which quarterly sales figures are required.

Sorting Statistics

Field definition

Short name	Y		
Field description	Year		
Heading	Year		
Functional group	Flights		

Attributes

☐ Same attributes as field
☒ Text field No. of characters: 4
☐ Calculation field Number of digits: Decimal places:

☐ Date field
☐ Time field
☐ Symbol
☐ Icons

Calculation formula

☐
 Condition:
☒ Input on selection screen ☒ Mandatory

☒ Complex calculation Fields Symbols

There are two things to take into account with the fields for quarterly sales figures. The InfoSet retrieves per flight (flight date) the total existing bookings. Using the flight date, you can thus determine to which quarter this total is to be assigned. Furthermore, only the quarters of a pregiven year should be determined according to the task. A quarterly sales figure is, therefore, the same as the booking total if the flight data lies in the pregiven year and in the correct quarter. As an example, we can use the definition of the field for bookings in the 2nd quarter.

Field definition

Short name: Q2

Field description: Bookings for the 2nd quarter

Heading: 2nd quarter &Y

Functional group: Flights

Attributes

☒ Same attributes as field

☐ Text field No. of characters:

☐ Calculation field Number of digits: Decimal places:

☐ Date field

☐ Time field

☐ Symbol

☐ Icons

Calculation formula

☒ PAYMENT

Condition: FD[YEAR]=Y AND 4<=FD[MONTH] AND FD[MONTH]<=6

☐ Input on selection screen ☐ Mandatory

☒ Complex calculation Fields Symbols

Please note that this example uses a replacement variable (&Y) in the column header.

The definition of the statistic itself is simple:

Title Quarterly revenues

Define statistic

Field	No	Srt	De	Su	NS	No	Av	%	Len	Rnd	Unit	Text
Airline carrier Id	1	1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>							<input checked="" type="checkbox"/>
Bookings for the 1st quarter	3		<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	18		DEM	
Bookings for the 2nd quarter	4		<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	18		DEM	
Bookings for the 3rd quarter	5		<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	18		DEM	
Bookings for the 4th quarter	6		<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	18		DEM	

Defining Headers in Statistics

Defining Headers in Statistics

On the *Statistic Header* screen, you can define your own headers for the statistic. To do this, you choose *Next screen* to go to the next screen in the sequence.

Page header (headers and column headers)

Quarterly revenues for &Y				
Air	1st quarter &Y DEM	2nd quarter &Y DEM	3rd quarter &Y DEM	4th quarter &Y DEM

Line structure

Air	Bookings_for_the_1	Bookings_for_the_2	Bookings_for_the_3	Bookings_for_the_4
-----	--------------------	--------------------	--------------------	--------------------

Page footer

--

In general terms, maintaining headers for statistics is exactly the same as for basic lists (see [Changing Headers \[Seite 158\]](#)).

The maintenance screen shows the basic structure of the statistic. It contains the header lines followed by the line structure and then the footer lines.

Fixed header lines depend on the structure of the query. These header lines can accept input on the screen and you can maintain them simply by entering your own values. The *Edit* menu contains functions which allow you to insert or delete lines.

At least one fixed header line is available with statistics. The statistics title is output in this header line.

You can define fixed header lines and footer lines so that when you generate the query list, they can receive current values of certain fields.

You maintain column headers field by field. If you do not want to use the header proposed by the InfoSet, you can define a query-specific header for each field. If you want to change a header for a particular column, place the cursor on the relevant field in the column header area and select *Edit* → *Column header* → *Maintain*.

With currency amount fields and quantity fields, you always use the second line of the column header for specifying the reference unit. When entering your own headers, you should therefore ensure that the headers only cover one line.



You can also set graphics parameters for each statistic. This works in exactly the same way as for basic lists (see [Graphics \[Seite 161\]](#)).

Error Analysis with Conversion Errors

Using quantity fields can sometimes result in conversion errors, since quantities are not easy to convert from one to the other.

At the end of the query list, there is an overview of all the conversions which have been performed. In the case of conversion errors, the table also indicates in which statistic and in which field the error occurred. Other possible causes of error are:

- no exchange rate found (with currency amounts)
- unknown unit (with quantities)
- dimension (with quantities, no conversion possible)

In the statistic itself, values which cause a conversion error are highlighted.

Defining Ranked Lists

This section describes how ranked lists are used and how they can be created.

What are Ranked Lists?

In the last section you familiarized yourself with statistics. With statistics, numerical values (for example, sales) belonging to particular key terms (for example, an airline carrier or a charter flight) are summed. The result is displayed in a table and thus returns an overview of the distribution of numeric values across the individual key terms.

Ranked lists are special types of statistics. Here, numeric values are also summed for key terms and displayed in a table. Sorting always takes place using a numerical value. This value is described as a ranked list criterion. Additionally, only a certain number of items are output. Ranked lists are therefore useful for asking questions such as: "Which 10 flight connections have the highest sales?"



When you choose a numeric value as the only sort criterion in a statistic, the result is practically a ranked list. With statistics, however, you cannot restrict the number of items to be output.

Creating Ranked Lists

Creating Ranked Lists

We can explain the definition of a ranked list by means of a simple example. Suppose you want to find out the 10 flights which have the most free seats.

Free seats per flight					
Sequence	Flg	Nr	Fl. date	Free	Max. cap.
1.	AA	0017	19.11.1999	660	660
2.	AA	0017	21.12.1999	650	660
3.	AA	0017	19.12.1999	643	660
4.	AA	0017	29.11.1999	612	660
5.	AA	0017	22.11.1999	564	660
6.	LH	0400	29.12.1999	380	380
7.	UA	0941	29.12.1999	370	380
	LH	0402	22.11.1999	370	380
9.	LH	0402	19.11.1999	364	380
	UA	0941	09.12.1999	364	380

To define this list, you create a query R1 via the InfoSet FLBU. From the field groups *Flight connections* and *Flights*, you choose the fields *Airline carrier ID*, *Flight connection ID*, *Flight date*, *Free seats* and *Total of current bookings*. Then, you call the *Ranked list* function to branch to the definition of a ranked list.

Title	Free seats per flight						
No. of places	10						
Define ranked list							
Field	No	Crit	Asc	Len	Rnd	Unit	Text
Airline carrier Id	1						<input checked="" type="checkbox"/>
Flight connection Id	2						<input checked="" type="checkbox"/>
Flight date	3						<input checked="" type="checkbox"/>
Free seats	4	<input checked="" type="radio"/>	<input type="checkbox"/>				
Maximum capacity	5	<input type="radio"/>	<input type="checkbox"/>				
Total of current bookings		<input type="radio"/>	<input type="checkbox"/>				<input type="checkbox"/>

First, you must assign a title to the ranked list. This is necessary, since one query can contain several ranked lists. You can then specify how many entries (i.e. output lines) you want the ranked list to have. The proposed value here is always 10.



Entries in ranked lists with the same value for the ranked list criterion are always given the same ranked list number (first column). The ranked list number is only displayed at the first entry. In addition, more rankings may be displayed than are defined in the ranked list definition. This is the case when entries with the same ranked list criterion values occur at the end of the ranked list. Here, all entries found to fulfill this same criterion are displayed.

Subsequently, you must define which fields are to be output and in what order. To do this, enter sequence numbers in the first column. You select one of these fields as the ranked list criterion. In our example, it is the field *Free seats*.

The ranked list criterion is the sort criterion of the ranked list. The default sort sequence for the sort criterion is descending order, but ascending order is sometimes necessary. This would apply, for example, to vendor sales figures which are stored as negative amounts. In such cases, you can specify ascending order for the ranked list criterion by selecting the column *Asc*.

You can also specify an output length and a rounding factor. As with statistics, currency fields and quantity fields require you to specify a reference currency or a reference unit which is used to convert all the amounts. Conversion errors also have the same procedure as statistics (see [Error Analysis with Conversion Factors \[Seite 189\]](#)).

All ranked lists contain lead columns. They are determined by SAP Query itself. The color attribute *Key* is automatically assigned to all non-numeric fields. When you scroll horizontally, these fields always remain in the visible part of the screen.

As with statistic, you can access the screens for maintaining the header and setting graphics parameters by following the normal sequence of query screens. The same rules apply for both these screens.

If you want to define another ranked list, you choose the *Next ranked list* function.

Changing Queries

Changing Queries

To change an existing query, proceed as follows:

1. Choose the component *Maintain Queries*.
2. On the initial screen, choose a user group and enter the query name. The query you specify must already exist in the user group.
3. Choose *Change*.

This takes you to the *Title, Format* screen. You then proceed as described above.

If you want to know who created the query or who last made changes, choose *Extras* → *Status info...*

Maintaining Queries in the Human Resources Application (HR)

When you create queries via InfoSets which use logical databases from the Human Resources (HR) application (for example PNP or PCH), you should be aware of certain special features on the *List Line Output Options* screen. The following sections explain these features:

Line Groups

Line Groups

On the *List Line Output Options* screen, you can group different lines together to form line groups. Lines which belong to a common line group are always output together as a block.



Suppose a person has several addresses. Usually, the name of the person is output on the first line. The second line contains the house number and street, and the third line displays the postal code and the city. By grouping lines two and three together to form a line group, you ensure that they are always treated as a block and output at the same time. Otherwise, for a person with more than one address, you would get a list with all the streets and then all the cities.

The example below shows the formatting of an address list in the application HR:

List line output options

Line No.	Color Gr	Header line	Ref.	Slash Bef / To	Blank line Bef / Aft	Columns with	Page header	New Page
1	1	List line	1	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	2	List line		<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	2	List line		<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Line structure

```

No.  ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+
1    |First_Name_____
1    |Last_Name_____
2    |House_Number_and_Street_____
3    |Postal_Cod City_____
  
```

With HR queries, you must also remember that the output in the column *Ref.* on this screen always refers to line groups. Thus, if you want to output lines only under certain conditions, you must define line groups.



The assignment of lines to line groups must be complete. This means that you must either assign all lines or to line groups or assign none at all. In the latter case, however, all lines are considered as belonging to a line group.

Evaluations According to a Company's Organizational Structure

Evaluations According to a Company's Organizational Structure

To generate evaluations of HR master data according to a company's organizational structure in the HR Planning area, you must use an InfoSet from the database PCH. This contains info types from HR master data and HR Planning, for example 0001 - 0007 and 1000 - 1002.

To clarify the structure, the HR Planning application area includes additional fields for the info type *Object name*. These fields use periods to show the object levels.

You want to create a query that gives a list of employee addresses per organizational unit. Select the relevant fields and arrange as follows:

List line output options

Line No.	Color Gr	Header line	Ref.	Slash Bef / To	Blank line Bef / Aft	Columns with	Page header	New Page
1	1	List line	<input checked="" type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	2	List line	<input checked="" type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	3	List line	<input checked="" type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	3	List line	<input checked="" type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Line structure

```

No.  ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+
1    |Object_ID_With_Lev Object_Abbreviation_Wi      |
1    |Object_Name_With_Level_Display_____          |
2    |First_Name_____                              |
3    |House_Number_and_Street_____                  |
4    |Postal_Cod City_____                          |
  
```

In order to make the list easier to read, you are recommended not to output lines 2,3 and 4 until column 15.

Evaluations According to a Company's Organizational Structure

When you execute the query, you see the selection screen for HR Planning. You have to execute the query with an evaluation path that evaluates associated people. SAP delivers the evaluation path 'person' that you could use here. You specify all the other parameters as you would for normal HR Planning reports.

The query then produces the list.



You cannot specify a sort sequence with analyses based on a company's internal organization.



You cannot evaluate factoring in a query.

When you select persons and info type records, the conditions of the underlying logical database apply. The time period selection refers only to the validity period of the info type records. If the beginning or end of the validity period falls within the selected period, the length of the validity is given when you output the data. In other cases, the period selected is displayed.

System Administration

The following sections are directed towards the system administrator. Here you will learn how to create work environments for the end user.

Managing InfoSets

This chapter describes why InfoSets are useful and shows how to gain an overview of existing InfoSets and how to manage them.

SAP Query allows you to evaluate data in the R/3 System. But since the R/3 System contains several hundred thousand fields in logical databases, tables and sequential datasets, it is simply not practical to offer all these fields for selection when creating queries. Therefore, before starting to create queries (using the component *Maintain Queries*), you create InfoSets (using the component *Maintain InfoSets*). Functional areas provide the user with a framework for defining a query quickly and without difficulty.

When you create an InfoSet, you select a logical database from an application system. However, since one logical database can still contain a very large number of fields, you combine fields together in logical units known as field groups.

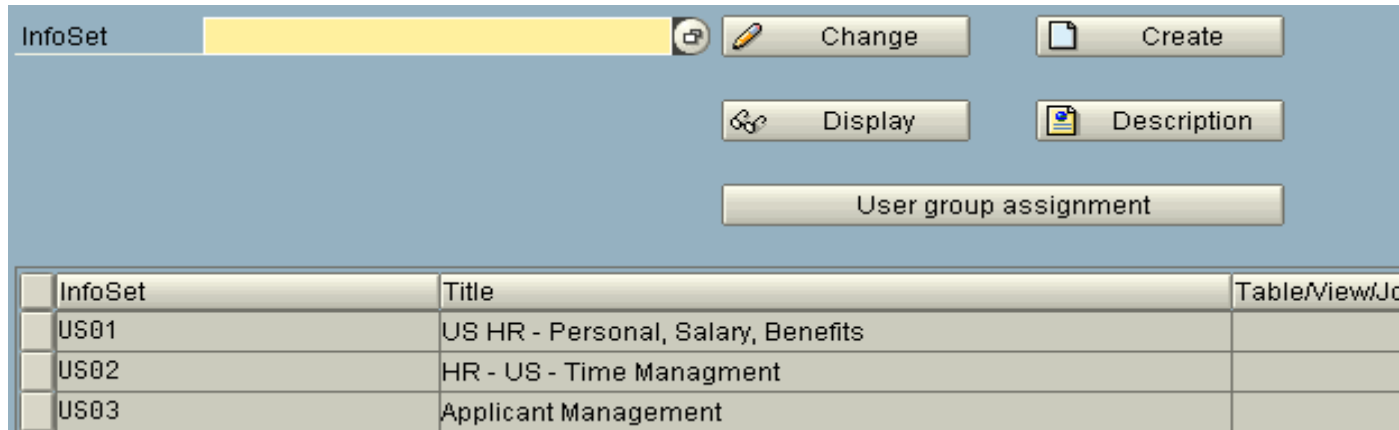
An InfoSet not only allows you to restrict the number of fields and group them together in meaningful units, but also to define auxiliary fields and then process them like database fields. You can also read the long texts in additional tables (for example the long text of an airline carrier in the table SCARR) and perform any necessary preliminary work. This means that you can evaluate sequential datasets just as easily as SAP databases. For further information, refer to [Creating and Changing InfoSets \[Seite 218\]](#).

The initial screen of the component *Maintain InfoSets* offers you a range of functions for maintaining InfoSets.

Selecting InfoSets for Processing

Selecting InfoSets for Processing

This is what the initial screen for the InfoSet maintenance component looks like:



InfoSet	Title	Table/View/Join
US01	US HR - Personal, Salary, Benefits	
US02	HR - US - Time Management	
US03	Applicant Management	

When you start the component, the system displays the names and titles of the InfoSets that have already been defined in the lower part of the screen.

A range of display and maintenance functions is available to you for these InfoSets.

There are two possible ways of selecting an InfoSet for processing by one of these functions:

- Enter the name of the InfoSet in the appropriate input field, or
- Select the list entry for the InfoSet in the lower part of the screen.

You can now execute the function.



Whenever you call value help for the *InfoSet* input field, a list of InfoSets that have already been defined is displayed. There are several different functions available:

Choose the *Search* function if you are looking for particular terms. In the directory, you can use the *Database structure* function to retrieve detailed information about the logical database underlying an InfoSet. This information includes specifications about the database structure as well as any select options and parameters that appear on the logical database selection screen.

You can choose *Extras* → *Status info...* to display information about a particular InfoSet including who its author is and who altered it last. Up to 10 previous alterations are displayed with their date and the name of the user who made the changes.

Displaying the InfoSet Directory

This section describes the functions for displaying InfoSets and associated queries:

Directory of Queries for an InfoSet

To display the directory of queries defined for the specified InfoSet, choose *Environment* → *Directories* → *Queries for InfoSets*.

You then see the names of the user groups as well as the name and long text of the queries.

Query Directory Functions

With *Goto* → *Query Directory*, you start a report which you can operate in the query directory. You can control the resetting of change locks, the generation of query reports and the deletion of queries from the directory. In any of these cases, the report selection screen allows you to restrict the section of the query directory shown to the relevant extracts.



You can search for a particular query which is locked against changes by other users or for queries locked by a particular user.

The following example shows such a query directory.

Functional area FLBU		
User group	Query	Records for sa
FB	G1	Flight connections (1st example)
	G2	Flight connections (2nd example)
	G3	Flight connections (3rd example)
	G4	Flight connections (4th example)
	G5	Flights (sorted by flight connection)
	G6	Flight bookings
	G7	Totals (sales per airline/connection)
	G8	Flight connections (alternate display)
	R1	Descending list of free seats
	S1	Sales per airline
FR	S2	Sales per airline and flight connection
	S3	Quarterly sales per airline for one year
	00	Contents of F1/S (without additional tables)
	01	Contents of F1/S (with additional tables)
	02	Contents of F1/S (with local fields)
	03	Quarterly sales (with local fields)
	04	Flight connections

You can see from the above list that there is a checkbox for each query. Select the checkboxes of the queries you want to delete and then choose *Delete sel. queries*. Before deleting a query, the system always requests confirmation. Afterwards, it refreshes the query list.

You proceed in exactly the same way, if you reset the change locks of queries or want to force the generation of query reports.



It can be useful to generate query reports after making changes in an InfoSet, if you want to ensure that the changes have really been adopted in the queries.



The *Generate sel. queries* function does not regenerate query reports, but sets a flag to indicate that the report is to be regenerated before the query is next executed.

For each query, the list also indicates how many records of background memory (file AQDB) are occupied by the saved lists of this query. Each record contains about 2900 bytes.

Saved Lists Directory

Saved Lists Directory

After calling the *Goto → List directory* function in the initial screen of the *InfoSet maintenance* component, an overview is displayed showing the total number of records per user group that are occupied by saved lists belonging to the queries of this user group. Each record contains about 2900 bytes.

Overview		
User group		Records for saved lists
MI	Master data	4
MMTEST	Materials Management, Local	9
RX	Personnel planning	13
Total records for saved lists		26

A user group can be selected for closer examination from this overview. You will then see another overview which shows the total number of records containing saved lists for each query in the user group. From this overview a specific query can then be selected for closer examination. The individual saved lists are displayed together with their creation date and time, long text and the number of data records used. This is in fact the same screen as that displayed when the *Goto → Saved lists* function is called for a specific query in the *Query maintenance* component. It is also possible to delete a list from this screen.

You can find further information on working with lists in [Saving and Redisplaying Lists \[Seite 113\]](#).

Maintaining InfoSets

This section describes the functions for maintaining InfoSets.

Displaying InfoSets

To display information about the structure of an InfoSet, first select the name of the InfoSet you are interested in. Now you can either choose *Description* or *Display*. If you want to see all of the details for the InfoSet definition, choose *Display*. All screens are displayed in the same manner as if the InfoSet were being changed, however no new entries can be made.

By selecting *Description* on the initial screen, you can display all the relevant information about an existing InfoSet.

The resulting list consists of a header common to all InfoSets and one or two sections containing information about each version of the InfoSet, which exists at any one time.

There may be two versions - a generated version and a corrected version. For each of these versions, the list contains information about the InfoSets and the fields assigned to them as well as about other components. For further information, refer to [Creating and Changing InfoSets](#) [Seite 218].

Copying InfoSets

If you want to copy an InfoSet, choose the *Copy...* function on the initial screen.

In the dialog box, enter the name of the InfoSet you want to copy. The InfoSet currently being processed is the InfoSet that the system proposes. You can overwrite this proposal. Then, enter the name you want to give the copy and proceed with the copying process. The copy is saved automatically.



SAP Query always copies the generated version of an InfoSet. If no generated version of the InfoSet exists, you are requested to generate the InfoSet before copying.

Renaming InfoSets

Renaming InfoSets

To rename an InfoSet, select the *Rename* function on the InfoSet maintenance initial screen. In the dialog box, enter the name of the InfoSet that you want to rename. The InfoSet currently being processed is the InfoSet that the system proposes. Enter the new name for the InfoSet in the lower entry field.

The following objects have to be locked before this function can be executed:

- User group catalog
- InfoSet catalog
- Query catalogs for user groups to which the InfoSet you are renaming is allocated
- Queries with this InfoSet.

The function is executed only if you were able to set all these locks. As long as these locks remain set, other users may be prevented from continuing in their work. Renaming an InfoSet can be very time-consuming!

Deleting InfoSets

To delete an InfoSet, enter the InfoSet name on the initial screen and choose the *Delete* function. Confirm that you want to delete the InfoSet on the next screen that appears.

The system then checks to see if any queries exist for that InfoSet. If queries do exist, then the system asks you if you want to delete them. If you confirm this, you go to the list of queries to be deleted. The InfoSet itself cannot be deleted until no more queries are attached to it.



If you delete an InfoSet, you also delete it from the user groups to which it is assigned. Since you cannot lock user groups for this purpose, InfoSets can be deleted only if the user groups are not being edited by another user (see [Functions for Managing User Groups \[Seite 285\]](#)).

Saving InfoSets

Saving InfoSets

When you save an InfoSet, all the information and properties for the InfoSet are written to the query database. These are stored in the query database as a corrected version of the InfoSet. If a corrected version already exists, it is overwritten.

If you want to change an InfoSet, the system always offers you the corrected version, if one exists in query database. If no corrected version is available, it retrieves the generated version.

You can use the **Save** function on most screens for processing InfoSets. It is not available on the initial screen, in dialog boxes, or in lists.

Generating InfoSets

If you want to store a generated version of an InfoSet in the query database, use the *Generate* function.

The existence of a generated InfoSet version is a prerequisite for

- assigning user groups to the InfoSet,
- creating queries for the InfoSet,
- copying the InfoSet and
- transporting the InfoSet.

The generated version is derived from the corrected version. If the generation is successful, the system deletes the corrected version from the query database.

When generating an InfoSet, the system produces a log. From this, you can check whether there are any errors or warning messages. The generation process involves the following steps:

- Comparison with the ABAP Dictionary

If, for example, you include a field from a logical database table in an InfoSet field group when you are processing the InfoSet, or you establish a link to an additional table, the system transfers the field or table field properties from the ABAP Dictionary to the InfoSet at the point it is created. When you save for the first time, these properties become an integral part of the corrected InfoSet version. During the generation process, the system performs a comparison with the ABAP Dictionary and reports any differences detected.

If there are differences such as changes in the structure of a table defined in the ABAP Dictionary, the system does not automatically modify the InfoSet, since it is not always possible - and meaningful - to do this. Deviations are reported in detail in the log list. For more information see the section on [Adjusting InfoSets \[Seite 217\]](#).

- Global Syntax Check

Several sections of ABAP code can belong to an InfoSet. When you create a query for an InfoSet, the system places these sections of code in the query report according to fixed rules (for example, depending on the hierarchy level of a logical database table).

The global syntax check determines whether all the InfoSet components contain syntax errors in any of the query reports. A successful InfoSet syntax check is an absolute prerequisite for running query reports.

Although SAP Query automatically performs a syntax check when you maintain an InfoSet which contains a section of ABAP code (for example, in the case of an additional field), this check can only verify the syntax of the code for this InfoSet component within its environment at the time of maintenance.



If the global syntax check detects errors, you can use the generation log to determine which InfoSet components contain ABAP code that could result in syntactically incorrect query reports. You must eliminate these errors if the generation is to run successfully.



You can use the function *Goto → More functions → Check and generate* to check and generate more than one InfoSet at a time. Initially, this function calls a selection screen where you can enter the InfoSets you want to check and generate. In addition, you must designate whether these InfoSets are simply to be checked, or if they should be generated as well. The function generates a log as part of its results,

Generating InfoSets

listing the errors found or informing you that the checking and generation process has been successfully completed.

Assigning InfoSets to User Groups

You can only define queries for those InfoSets assigned to your current user group. On the initial screen of the component *Maintain InfoSets*, you can use the *Assign to user groups* function to specify which user groups you want to assign to a particular InfoSet.

To assign an InfoSet, proceed as follows:

1. Enter the InfoSet name in the field *InfoSet* on the initial screen.
2. Select *Assign to user groups*.

This generates a list of all the user groups. You can assign the InfoSet to the user groups you want by selecting.



You can also use the *Assign to user group* function if you have already assigned the InfoSet to some user groups, but you now want to change the assignment. The user groups to which the InfoSet has already been assigned are flagged.

Checking InfoSets

Checking InfoSets

The *Check* function performs the global syntax check described above but does not create a generated version in the query database. When maintaining an InfoSet component which contains ABAP code, you can also perform a global syntax check in addition to the automatic local syntax check.



You can use the function *Goto → More functions → Check and generate* to check and generate more than one InfoSet at a time. Initially, this function calls a selection screen where you can enter the InfoSets you want to check and generate. In addition, you must designate whether these InfoSets are simply to be checked, or if they should be generated as well. The function generates a log as part of its results, listing the errors found or informing you that the checking and generation process has been successfully completed.

Adjusting InfoSets

Use *InfoSet* → *More functions* → *Adjust* to find out if the technical properties of fields from database tables are used in a particular InfoSet have been changed (for example, type, length, output length, use as a currency amount or quantity field). Changes like this should, however, be the exception.

If, however, these kinds of changes have been made, a warning is displayed when the InfoSet is generated in which the differences between fields in the InfoSet and Dictionary fields are described. *InfoSet* → *More functions* → *Adjust* allows you to copy these technical property changes for your InfoSet if you so wish.

Initially, the screen is displayed with the same warning that appeared when it was generated. This screen also contains an *Adjust* function that you can use to copy the appropriate Dictionary entries for your InfoSet.



Immediately after 'adjusting' an InfoSet, all queries that use this InfoSet should also be 'adjusted' so that all new properties can be changed within each query's definition.

Creating and Changing InfoSets

The following sections explain how to create and change InfoSets. The information is intended primarily for system administrators.

[Creating InfoSets \[Seite 219\]](#)

[Assign Data Source \[Seite 220\]](#)

[InfoSet Display \[Seite 223\]](#)

[Definition of Field Groups \[Seite 226\]](#)

[Obtaining Additional Information \[Seite 231\]](#)

[Creating Selections \[Seite 250\]](#)

[Further Codes \[Seite 257\]](#)

[Application-specific Enhancements \[Seite 264\]](#)

Creating InfoSets

If you want to create an InfoSet, you should first discuss the requirements for the InfoSet with the appropriate application department.

You must clarify the following points:

- Which data source corresponds to the requirements?
Example: Material evaluations in materials management or document evaluation in financial accounting.
- Which fields do you need to include in the InfoSet?
- Do you need additional information which is not available in the data source? (This calls for the linking of additional tables and the definition of additional fields).
- Do you need parameters and selection criteria? (parameters and selection criteria appear on the selection screen of queries which are created using the InfoSet).
- Do you have to take any special measures such as access protection?
- Do you have to change the long texts and headers of the selected fields?

The last two points are rather less important, since you can add these components to an existing InfoSet at any time. However, the choice of data source determines the reports you can produce with this InfoSet.

When you have resolved these questions, you can create the InfoSet. Proceed as follows:

1. Call the component *Maintain InfoSets*.
This takes you to the initial screen for editing InfoSets.
2. Enter a name in the *InfoSet* field.
InfoSet names can contain up to 24 characters. The lower half of the screen displays all of the InfoSets that already exist.
3. Choose *Create*.

You then go to the *Title and Database* screen. For further information, read the following sections.

Assigning Data Sources

Assigning Data Sources

On the *Title and Database* screen, you enter a description of the InfoSet. The end user sees this name when creating a query. It should, therefore, allow for an InfoSet to be identified.

The field *Authorization group* allows you to enter an authorization group up to eight characters long. If you specify an authorization group, it is inherited as an attribute by every report generated for a query via this InfoSet. This means that only users authorized in their user master records to execute programs from this authorization group can execute these queries. Authorization groups thus provide a way of protecting reports generated by the query from unauthorized execution.






Please do not confuse authorization groups with user groups.

Authorization groups can be assigned for any reports and determine a user's right to execute a program. Authorization groups can be entered in user master records using the authorization object S_PROGRAM. (You can find more detailed information on this authorization object in transaction AUTH_DISPLAY_OBJECTS. Choose here the authorization class *Basis Development Environment* and then the documentation for the authorization object S_PROGRAM).

User groups help you to control access and change authorizations for queries in the SAP Query environment.

The following five options are open to you when selecting the data source:

Table join using Basis table	<p>A table join always takes in several tables that are read with the aid of special forms of the SELECT instructions. Enter in the entry field the first table in the join. The additional tables can be entered after choosing the <i>Continue</i> pushbutton.</p> <p>You can find more information in Table Join Definition Procedures [Seite 272].</p>
Read table directly	<p>The data is read directly from an SAP table. The table must be entered in the ABAP Dictionary and must exist in the database. Further information can be found under Reading Directly [Seite 277].</p>
Logical database	<p>Enter the name of the logical database. The input help allows you to get a list of all logical databases. From this list, you can request detailed information about individual logical databases. The logical database structure, that is the hierarchical relationship of logical database tables to each other, is particularly important for creating InfoSets.</p> <p></p> <p>Pay attention to the remarks contained in the appendix: Logical Databases with Different Node Types [Seite 346].</p> <p>If you have already entered a logical database, you can request detailed information about it by using the  function.</p> <p>In the <i>Selection screen version</i> field, you can enter an ID for the version of the selection screen you want to use for queries in this InfoSet. If you leave the field blank, the system calls the standard selection screen when the query is executed. The version consists of a three-character ID. The chosen selection screen version must be supported by the logical database. You can use the <i>Possible values</i> help to find which selection screen versions are available.</p>
Retrieving data with programs	<p>Using an ABAP Report you can make evaluations on datasets, for which the automatic data retrieval of the query is not sufficient. You can find more information under Data Retrieval by a Program [Seite 279]</p>
Sequential Datasets	<p>Select this field if you wish to allow evaluations on a sequential dataset, without having to take care of the data retrieval in query reports yourself. Then, enter the sequential dataset file name you want to read. Its record structure should correspond to the structure specified. Further information can be found under Sequential Dataset [Seite 278].</p> <p></p> <p>The option <i>Sequential Dataset</i> is shown under <i>Data Retrieval by Program</i> if you choose the pushbutton <i>Additional Options</i> in the dialog box displayed above.</p>

If a text field exists for a particular field then this is displayed in the InfoSet maintenance with a corresponding icon. When creating queries using the InfoSet, you can choose between

Assigning Data Sources

displaying the value and displaying the relevant text in the report. Choose the option *No automatic text recognition* if this is not required.

You can use the field *Fixed point arithmetic* to specify whether queries defined via this InfoSet should work with fixed point arithmetic switched on or off. The contents of this field are thus an attribute of the generated reports. In the standard system, fixed point arithmetic is always switched on.



You can change the fixed point arithmetic status for an existing InfoSet. When doing this, however, you must check all the code relating to the InfoSet. If this code contains calculations with values of type P, you must adapt these calculations to the modified fixed point arithmetic status.

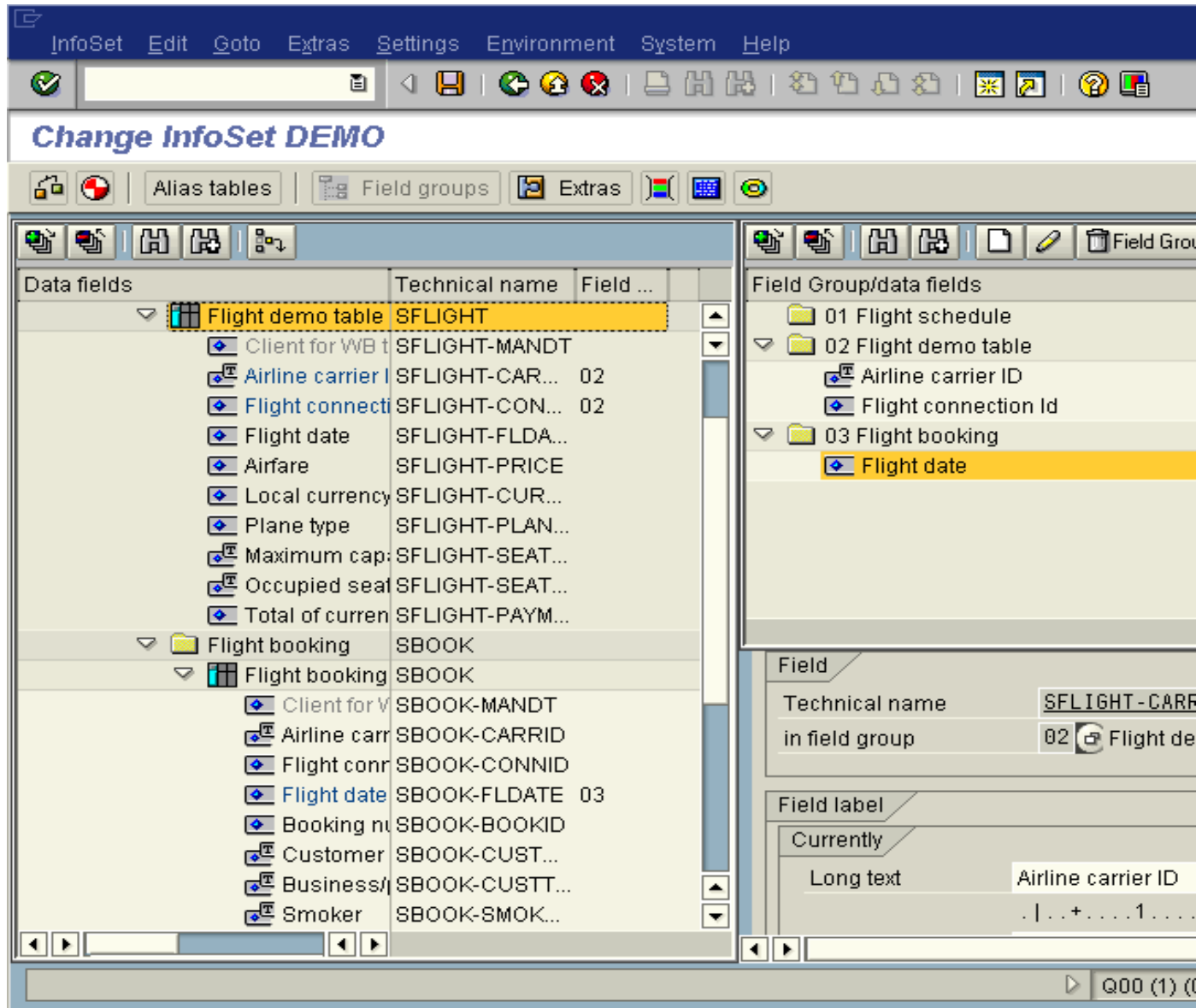
If you choose the pushbutton *Additional Options* then the entry fields *Class for Text Identification* and *InfoSet Generator* appear. Compare here the actions in the section [Application-specific Enhancements \[Seite 264\]](#).

The example in this section refer to InfoSets using the logical database F1S (flight bookings).

InfoSets using a logical database and InfoSets without any underlying database are, to a large extent, made in the same way so that the actions in the following units both refer to logical databases as well as to other options for retrieving data. For this, read the section entitled [Special Features \[Seite 282\]](#) under *InfoSets without a Logical Database*, or the actions in the sections that are referred to in the table above.

InfoSet Display

You can see the InfoSet Display on the following screen.



If you select *Change* on the initial screen, the screen above is subsequently displayed. Although you can still return to the title screen by choosing *Goto* → *Title, database*, you can only make changes which do not affect the structure of the InfoSet data.

The data source structure (logical database, table join, table) with the corresponding enhancements (additional tables, additional structures, additional fields) is displayed as a tree in the left half of the screen. Each field is marked with an icon. The icons *Simple Field* and *Field*

InfoSet Display

with *Text* are possible. In the first case, a field can only be provided with its own values. In the second case, a text can be automatically created for each field value. You can find more information under [Text fields \[Seite 225\]](#). Two columns containing the technical names and assignment for a field group are assigned to each field.

With logical databases this tree mirrors the logical database's structural tree and contains all its tables along with the hierarchical relationships that have been defined between them. With InfoSets using logical databases from the HR application, the tree contains all info types that have been copied into the InfoSet by the InfoSet Generator. These information types also appear next to each other on the same level.

The fields defined by the dictionary for a certain table, as well as additional fields that have been explicitly assigned to the table, hang underneath every table.






- Fields from an additional table assigned to the database table (see [Assigning Additional Tables \[Seite 234\]](#)).
- Additional fields assigned to the database table (see [Creating Additional Fields \[Seite 241\]](#)).
- Additional structures (see [Creating Additional Structures \[Seite 247\]](#)).

These fields appear behind the actual table fields and represent an enhancement of the original table.

The field groups are displayed as a tree in the upper right part of the screen. The same icons as found in the data source (left) and the technical names are assigned to each field in a separate column. The tree display makes it possible here, as well we in the data source, to expand and collapse to the section you want to see on the screen.

Creating and changing field groups is possible using special functions that can be found as pushbuttons above the field groups. To select fields for field groups, first select the fields you want with a single mouse click in the data source. These fields can then be transferred using drag&drop into the field group of your choice. It is also possible to select the relevant field group with a single mouse click and call the function *Add Fields to Field Groups*.

Double-clicking on a field in the data source or in the field group displays all the technical information for this field in the lower right part of the screen. You can change long texts and headers as and when necessary here. It is possible to mark every numerical field as one that 'cannot be summed'.

When you call the functions  *Extras*,  (*Selection*),  (*Coding*) and  (*Enhancements*), corresponding maintenance screens are shown in the right part of the screen. With  *Field groups* you can display field group maintenance again. The following sections tell you about the options that are available to you when you use these functions.

Text Fields

Every data source field is indicated with an icon that tells you if it is a normal field or a text field. You can use different processes (check tables, fixed values, and so on) to automatically define texts to field values. When using a field like this in a Query, you can decide whether you want to use the value or the assigned text. For further information, see [InfoSet Display \[Seite 18\]](#).

During the InfoSet maintenance, the data source of each field is inspected to see whether a text can be provided for a text field or not. The result of this check is displayed as an icon in front of the field.



You can only use values for selection and output with these fields.



A text can be defined automatically for every possible value of the field.

If a text field is transferred to a field group, the InfoSet user can always use values and texts. If a field like this is removed from a field group you can use neither the value nor the text. This means that you cannot only transfer the text of the field to a field group, for example.

You can use the technique of automatically providing texts for InfoSets that already exist. To do this, you have to choose *InfoSet → More functions → Update Text fields* for each InfoSet from the initial screen of the transaction SQ02. Texts exist for all possible fields after the InfoSet is regenerated.

Under some circumstances, it may be that you do not require the automatic provision of texts. In this case, you can switch off the automatic text identification when you create the InfoSet, in the first dialog box by selecting the relevant checkbox. Otherwise, you can switch off the text identification function for an existing InfoSet by choosing *InfoSet → More functions → Delete text fields* from the initial screen of the transaction SQ02. This procedure only works if none of the text fields has been used in a query.

The standard procedure for identifying texts is relatively extensive and complicated. There will still be cases, though, for which this procedure is insufficient. It is therefore possible to use other procedures for text identification. You can find more information under [Application-specific Enhancements \[Seite 264\]](#).

Definition of Field Groups

Definition of Field Groups

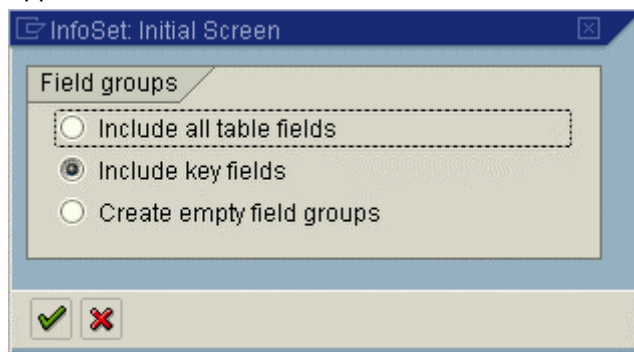
A field group combines related fields together into a meaningful unit. It provides you with a preselection, so that you do not have to search through all fields of a data source just to produce a simple list.

This means that the end-user only has access to precisely those fields assigned to a field group. Fields must be assigned to a field group before they can be used in queries.

Preassigning Field Groups

If you made your entries in the dialog box *Title and Database*, then field groups are prepared.

If the InfoSet is created using a single table or using a table join, then the following dialog box appears:



You can choose here which fields should be assigned to field groups.

If the InfoSet is created using a logical database then the following applies:

Where the logical database contains a maximum of four nodes, exactly one field group will be created for each node. If the logical database contains more than four nodes then you can determine first, in a dialog, for which nodes a field group should be prepared. For this, the structure tree of the logical database is displayed in a dialog box. Each node, that should have a field group prepared for it, can be selected here with a checkbox.

The prepared field groups have the same long texts as the corresponding nodes. They are empty, meaning that no fields have been assigned to them.



The logical database F1S consists of three hierarchically-arranged tables. Three field groups are generated in advance accordingly:

Logical database F1S		01 Flight schedule
Flight schedule	SPFLI	02 Flight demo table
Flight schedule	SPFLI	03 Flight booking
Flight demo table	SFLIGHT	
Flight demo table	SFLIGHT	
Flight booking	SBOOK	
Flight booking	SBOOK	

The predefined and perhaps preassigned field groups can be changed in any way you like. Long text can be changed. Field groups can be deleted or taken up afresh. You are recommended, however, to use and keep the preset 1:1 relationship between field groups and tables, or nodes.



The technology of retrieving predefined content has already been previously realized in the framework of InfoSet maintenance using the InfoSet generator for HR InfoSets (logical databases PNP, PCH and PAP). The existing generation shows a generalization of this technology. The InfoSet generator for HR InfoSets, however, still remains active, meaning that if InfoSets are created using logical databases PNP, PCH, or PAP, then this generator is used for retrieving the predefined content.

There are two different generators for InfoSets with this. You can also use additional generators, some can either be used generally for all InfoSets and others just for certain data sources. You can find information on connecting additional generators in the section [Application-specific Enhancements \[Seite 264\]](#).

For an InfoSet with a data retrieval program, a field group is created with all fields of the data structure.

Creating a New Field Group

To create a field group, proceed as follows:

1. Choose the symbol *Create* from the function toolbar (or function *Edit* → *Field Group* → *Create Field Group*). A window appears where you can define single or multiple new field groups.
2. Enter an abbreviation and a long text for each field group.
The abbreviation for a field group can consist of any character that you want. The abbreviation only serves a technical purpose within an InfoSet when fields are being assigned to field groups.
3. Choose *Continue*.

For further information, refer to the following sections.

Assigning Fields to a Field Group

Assigning Fields to a Field Group

You can assign fields of different tables to one field group. In this way, you can treat fields of linked additional tables, additional structures and additional fields in the same way as true logical database fields.

To assign a field to a field group, proceed as follows:

1. Click on the field you want to select in the data source.
 2. Use drag&drop to move the field into the field group of your choice.
- or;
1. Click on the field and on the field group of your choice.
 2. Choose the function *Add fields to field groups* or the relevant function from the context menu.

If you want to change the assignment of a field to a field group, click on the field to be changed and move it into the field group you want by using drag&drop, or double-click on the field and use the value help to change the field group in the screen in the bottom right-hand corner.

Double-clicking on a field in the data source or in a field group displays all the technical information for this field in the lower right part of the screen. You can change long texts and headers as and when necessary here.



The assignment between the field and the long text is not always unique, for example when several fields refer to the same data element. Therefore, you can overwrite all the field texts and should in any case always do this in the interests of the end-user.

There is a field header for every field. With fields defined in the ABAP Dictionary, the header text is used. The field headers are needed to determine the column headers for queries (see [Changing Headers \[Seite 158\]](#)). With fields defined in the ABAP Dictionary, you may want to replace the field header with a different text.

Headers can contain up to 30 characters. But they should not contain more characters than the field's output length, seeing as these will then be cut off during subsequent display of the query. A ruler is displayed above the header input fields allowing you to see how long your header can be. The output length of the field is also displayed.

If you have currency amount fields or quantity fields in an InfoSet, it is not necessary to include the associated currency or unit fields. The query takes care of the assignment between currency amount and currency fields, or quantity and unit fields itself.

You can use the functions *Find* or *Find next* to search for a text or technical name in both trees.



Client fields of client-specific tables are not included in the selection offered in the field groups, since cross-client reading is not supported by the query. If the value of the client field is required in special cases, an additional field has to be used. The client field can be accessed in the coding of an additional field, even if the client field is not part of the selection offered.

Deleting Field Groups/Fields from Field Groups

Deleting fields from a field group or an entire field group is easy if you have just created the InfoSet or, at least, not yet used it to define queries.

To delete a field group, place the cursor on the field group in the tree and call *Delete field group*. Deleting this field group means that the field assignments for this field group are lost.

To delete a single field from a field group, click on the field and choose *Delete field*.

In the left-hand tree you can select more than one field (with the control or shift button) and remove fields from the field group over the context menu.

If you want to change an InfoSet for which queries have already been defined, you must avoid inconsistencies between the InfoSet and existing queries. Adding individual fields or even entire field groups to the InfoSet of no significance for existing queries. Deleting individual fields or entire field groups means that queries requiring access to them can no longer be processed correctly. Therefore, you can only delete those fields and field groups that are not used in queries.

You can also determine in which queries a field is used. To do this, double-click on the field and choose the function *Environment* → *Directories* → *Queries for field* or choose *Queries for field* in the lower right-hand box. ABAP Query then displays all the user groups and queries that process the selected field.

If you still want to delete either a field from a field group or an entire field group, you must first delete or change all the queries affected by this. Otherwise, the system responds with an error message and refers you to the queries concerned.

Displaying a List of Assigned Fields

You can get an overview of the fields that are assigned to an InfoSet in the InfoSet maintenance screen by choosing the function *Expand All* in the right side of the screen.

If you choose *InfoSet* → *Description* on the InfoSet maintenance initial screen, you get a complete overview of the InfoSet, including a list of all field groups with assigned fields. You can also print this overview. You can also display a list of all field groups and assigned fields by choosing *Environment* → *Directories* → *Fields/field groups*.

If you are not sure whether to assign a field to a field group, you can request documentation. To do this, place the cursor on the field and choose *Extras* → *Field description*. You then see the field documentation. This also contains a technical description (type, length and so on).



This function is also contained in the lower right dialog box used for changing field texts.

Currency amounts are stored in the logical database as pairs of fields. The first field contains the amount and the second field the currency or unit. Although the fact that the logical database itself always ensures that the (amount, unit/currency) pair contains the correct values means that it is not necessary to know which reference fields belong to which amount fields when maintaining an InfoSet over a logical database, in the case of an InfoSet that is not based on a logical database, the InfoSet itself has to ensure that the values in the amount and unit fields belong together (for example, by reading another table). This means that the correspondence between the amount fields and unit fields has to be known.

The reference field is always displayed in the lower right box for changing field texts.

Obtaining Additional Information

In some cases, the information that a data source supplies is insufficient. In this way, for example, new information could be gained by simple calculations. If, for example, the maximum number of seats and the number of bookings for a particular flight are known, the number of unoccupied seats can be calculated easily. Such information can be useful for different queries evaluations.

Three options are open to you for obtaining additional information during InfoSet maintenance: additional tables, additional structures and additional fields

Additional tables read a record from a table. This table's fields can then be evaluated. The keywords that will allow you to choose the record must be provided by the data source.

Additional fields allow you to calculate new values from existing information. With this, additional fields have the limitation that they must have a simple (scalar) data type. Additional structures are actually additional fields with a non-scalar type of data. This data type must always be a (flat) structure from the ABAP Dictionary.

In every case it is important to choose the right time for either reading the record or performing the calculation. Here, the 'right time' stands for a table in the logical database. At this time all information necessary to read the record or run the calculation must be available to the system.

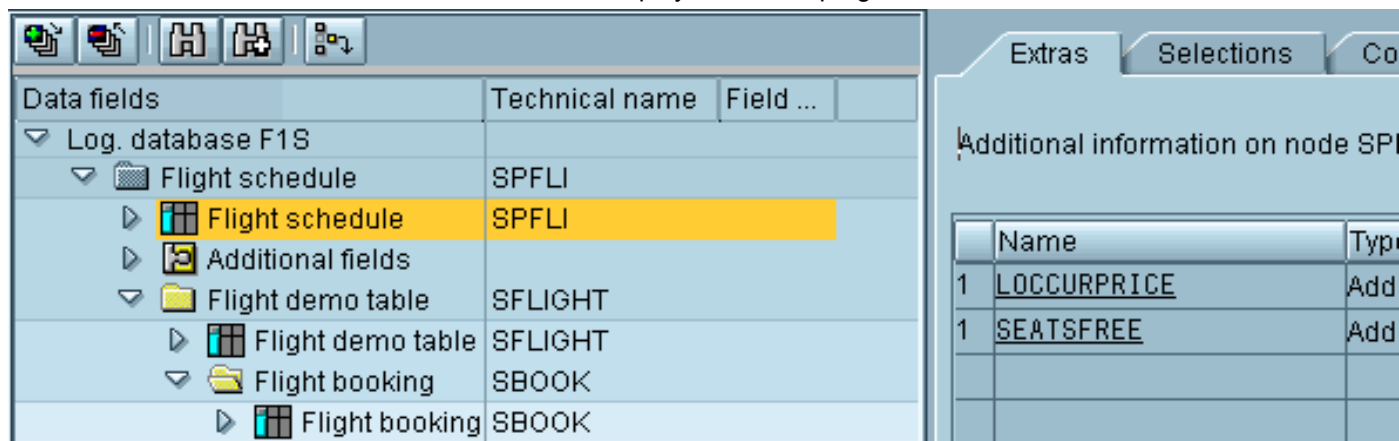
Obtaining additional information is thus very much dependent upon these 'times', that is upon the logical databases tables. You can assign as many additional tables, additional structures and additional fields to a logical database table as you want.

The fields from these additional tables and the additional fields themselves are retrieved when the logical database table is filled. Thus they are also contained in the field list that corresponds to the logical database table and can be assigned to a field group just like the table's fields.

Proceed as follows when creating or changing additional information:

1. On the screen for maintaining InfoSets, choose the table of the data source, for which you wish to maintain additional information, by double-clicking.
2. Choose *Extras*. Alternatively, you can use the context menu.

The additional information on the selected node is displayed in the top right window.



This window contains the name of the additional information, its type (additional table, additional structure, additional field, code) and a long text.

The window's first column contains a number. This number determines the order in which the code for each individual additional piece of information within the query is executed.

Obtaining Additional Information



The sequence numbers of the various individual pieces of additional information do not have to be different. You can find more information about sequence numbers in [Creating Additional Fields \[Seite 241\]](#)

This window also allows you to *Define*, *Create*, *Change* or *Delete* additional information. When using *Change* and *Delete*, you must place your cursor on the appropriate additional information entry. If you are trying to change an entry's attributes, double-click on it to branch to the additional information maintenance screen.

When connecting additional tables, additional fields and additional structures to logical databases, you must bear in mind the following points:

- Queries read linked additional tables using the ABAP statement
SELECT SINGLE * FROM <table> WHERE...
- You have to specify when to read the additional table.
You determine this by assigning the additional table to a particular logical database table. For example, it makes no sense to assign the table SCUSTOM (customer data) to the table SPFLI (flight connections) of the logical database F1S, since no meaningful values can be assigned to the key field SCUSTOM-ID (customer number) during the event that processes SPFLI data. Only when processing data from the database table SBOOK (bookings) can you use the customer number and then, for example, retrieve the customer name.
- You must fill all key fields.
- Finally, you must assign all additional table fields used in queries to a field group in the same way as table fields for a data source of a field group.
You can link any tables as additional tables, provided they are defined in the ABAP Dictionary and can be read with ABAP Open SQL.

Pay attention to the following when defining additional fields:

- You must specify when the calculation is to be performed.
You do this by assigning the calculation to a logical database table. It makes no sense to calculate the number of free seats on a flight from the fields SFLIGHT-SEATSMAX (maximum occupancy) and SFLIGHT-SEATSOCC (occupied seats) when processing the data of the table SPFLI because these fields only contain values at the level of processing for table SFLIGHT.
- You must specify the ABAP code for the field.
- You must assign the field to a field group.

Pay attention to the following when defining additional structures:

- Assign the structure to a node of the data source. You can find all individual fields from the additional structure in this node of the data source. Additional fields offer a simple way of calculating more than one additional field at the same time.
- Enter a calculation formula.
- You must assign the fields to a field group.

The following topics provide information on linking additional tables and the definition of additional structures and fields:

If you wish to provide an InfoSet with extras, that was not created using a logical database, then please take note of the actions in the following topics:

[Definition of a Table Joins \[Seite 274\]](#), [Reading Directly \[Seite 277\]](#), [Sequential Dataset \[Seite 278\]](#) and [Data Retrieval by a Program \[Seite 279\]](#)

Assigning Additional Tables

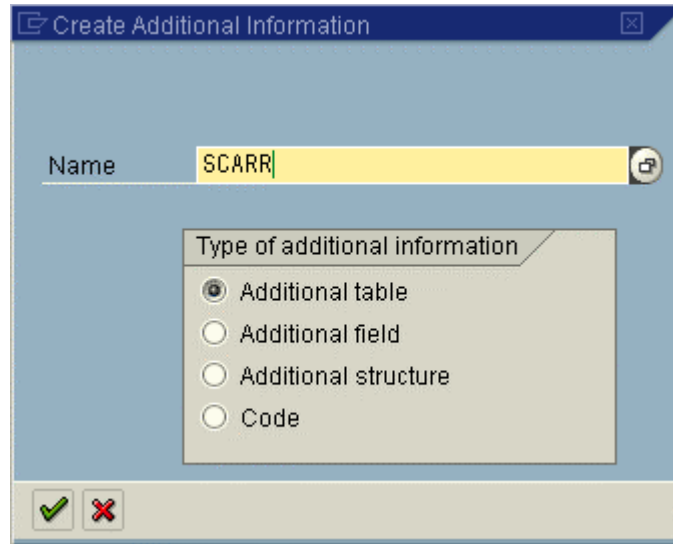
Assigning Additional Tables

To define an additional table for a data source table, proceed as follows:

Double-click on a table that you want to assign to an additional table and choose the function *Extras*.

1. Choose the function *Create* (in the lower right half of the screen).

A dialog box appears where you must first choose to define either an additional table, an additional field, an additional structure or code.



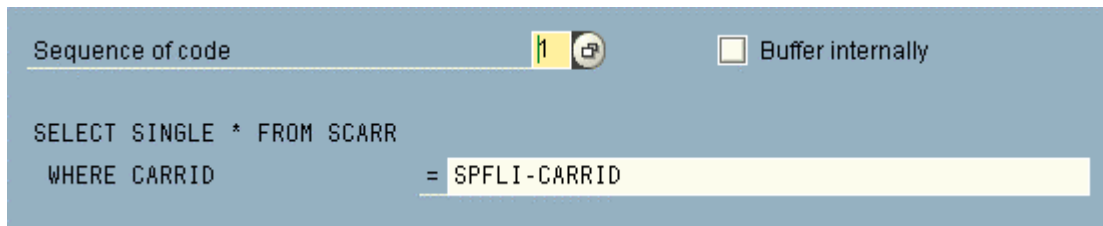
The *Change* function then automatically displays the appropriate maintenance screen. It can also be triggered by a simple double-click.

2. Choose *Additional table* and enter a name for the table. You can use all table names found in the ABAP Dictionary or those already declared as alias tables for your InfoSet (see *Multiple Additional Table Assignments* below).

The system then creates the corresponding SELECT statement in another dialog box, if you selected a logical database as a data source. It then proposes a range of the assignment of this SELECT statement in the code section of the generated query report for the GET event of the logical database table. This sequence is important where, for example, you use a table field that derives its value from a SELECT statement as a key in the SELECT statement of another table assigned to the same logical database table.

The system shows you the key fields for the specified table. The system often also gives you a proposal as to how the key field could be filled. For example, it always proposes SY-LANGU as the language key. However, such default values often serve no useful purpose and you must specify values for the key field yourself.

The following shows a SELECT statement using a default value:



Sequence of code 1 ☐ Buffer internally

SELECT SINGLE * FROM SCARR
WHERE CARRID = SPFLI-CARRID

Decide whether or not you want to use the *Buffer internally* option for the additional table.

This option can be used to improve the performance of your query. If you choose this option, then all queries that use this additional table will save the records they read from it in an internal table. If a record from the table is needed, the system first checks to see if the record already exists in the internal table. If the record does not exist in the internal table, then a SELECT SINGLE statement is used to read the record from the database and insert it into the internal table. Internal buffering prevents additional table records from being read more than once each time a query is executed.

Buffer internally should only be used when you expect an additional table record to be read numerous times during query processing. Otherwise you should avoid using this option because the internal table takes up memory space. In addition, with very large tables with numerous fields you should check to see if the time saved by using the internal table justifies the amount of memory space needed to store it.



In logical database F1S, which is used for examples in this handbook, two fields exist in the table SPFLI for departure and arrival airport. Both fields contain a key for the table SAIRPORT, which contains the long texts for airports. In a case like this, you should buffer the additional table internally, since it is safe to assume that records in SAIRPORT will be read multiple times.

You must now specify values for all key fields that do not contain a default value. You can, of course, overwrite the defaults.

For this purpose, you can use fields (for example SY-LANGU), number literals (for example 34), text literals (for example 'D') or parameters.

You should check carefully that any fields you assign to key fields contain valid values when the additional table is read. SAP Query helps you to do this by allowing only the following fields:

- Parameters (see [Creating Parameters \[Seite 252\]](#))
- Data from the DATA code (see [Further Code \[Seite 257\]](#)).
- Fields of the logical database table, to which the additional table is assigned, and fields from hierarchically superior tables of the logical database. In the case of the latter, any additional fields and additional table fields assigned there are also allowed.



If, for example, you assign the additional table to the table SFLIGHT of the logical database F1S, you can use fields from the tables SPFLI and SFLIGHT. It is also possible to access the fields of additional tables as well as additional fields assigned to the table SPFLI.

- Fields of another linked additional table assigned to the same logical database table.
In this case, you must ensure that SAP Query reads the additional tables in the correct sequence.
- Additional fields assigned to the same logical database table.

Assigning Additional Tables

Here, you must ensure that any calculations involving additional fields are performed before accessing the additional table.

You can choose *Check additional table* to check your SELECT statement for syntax errors. Otherwise it can only be corrected when you use the functions *Check* and *Generate* for the InfoSet as a whole.

Whenever you are on the screen for editing additional tables, you can change the following settings:

- the sequence of the SELECT within the processing of data for the logical database table.
- the *Buffer internally* option.
- the key field setting.

However, you cannot change the assignment to a logical database table. To assign an additional table to a different logical database table, you must delete it first and then assign it again.

Multiple Additional Table Assignments (Alias Tables)

The procedure described above allows you link a table to an InfoSet as an additional table exactly once. Whenever you attempt to link the table a second time as an additional table you receive an error message informing you that the table may only be used once. The following example demonstrates, however, that multiple linking of a table may be useful.



In logical database F1S, which is used for examples in this handbook, two fields exist in the table SPFLI for departure and arrival airport. Both fields contain a key for the table SAIRPORT, which contains the long texts for airports. If table SAIRPORT could only be linked to SPFLI as an additional table once, then only the long texts for one of these two types of airport entries could be accessed. The long text for the other airport entry would have to be determined in another manner (using an additional field).

A similar problem occurs when a table that you want to link as an additional table belongs to a logical database. Here you once again receive an error message stating that this table may only be used once.

You can, however, use alias tables to allow you to link your table more than once.

You can use several different alias names for a table. The table can be referred to more than once using the different names.

To define an alias name for a table, proceed as follows:

1. Choose *Alias Tables*.

A window appears displaying all existing alias names including the Dictionary table definitions to which the alias names refer. Alias names may not be repeated within an InfoSet. They must always refer directly to a Dictionary table definition and may not be identical to the name of any Dictionary definition.



The *Alias Tables* function also appears as a pushbutton in many dialog boxes.

2. In the lower left-hand corner of your screen choose *Create*.

Another window appears where you can enter the alias name and the name of the assigned table from the Dictionary.

In order to delete an alias name, place your cursor on the appropriate entry in the list of alias names and choose *Delete*. You can delete only if the alias table is not used in the InfoSet.

In order to link a table as an additional table within an InfoSet more than once, proceed as follows:

1. Link the table once using its original name.
2. Define an alias name for the table.
3. Link the table a second time as an additional table using this alias name.



Note that the fields from this table will now appear twice within the InfoSet with the same long texts and headers. Hence, when a table is linked multiple times, the long texts and headers of its fields must normally be changed whenever they are assigned to a single field group.

Assigning Table Fields to a Field Group

Assigning Table Fields to a Field Group

The fields for the inserted additional table are displayed at the end of the field list of the respective table, as follows.

The screenshot shows the SAP Query Designer interface. On the left, the 'Data fields' pane displays a tree structure under 'Log. database F1S'. The 'Flight schedule' folder is expanded, showing 'Flight schedule' (SPFLI), 'Airline' (SCARR), and 'Additional fields'. The 'Additional fields' folder is expanded, showing 'Local currency' (LOCCURPRICE) and 'Free seats' (SEATSFREE). The 'Flight demo table' folder is expanded, showing 'Flight demo table' (SFLIGHT) and 'Flight booking' (SBOOK). The 'Flight schedule' field is highlighted in yellow. On the right, the 'Extras' tab is selected, showing 'Additional information on node SPFLI'. Below this, a table lists the fields:

	Name	Type
1	LOCCURPRICE	Addi
1	SCARR	Addi
1	SEATSFREE	Addi

They can be assigned to a field group using the method already described.



Do not forget to assign the fields you need from a linked additional table to a field group. These fields can only be viewed by the end-user in the InfoSet and be used for defining queries after you have completed this assignment.

Displaying a List of Assigned Additional Tables

Otherwise, the system responds with an error message and refers you to the queries concerned. For each assigned additional table, the display lists the data source table assignment and the processing sequence for the table data. The subsequent lines contain the key field setting.

Deleting the Assignment of Additional Tables

Deleting the Assignment of Additional Tables

If you want to delete an additional table, you must first call the additional information screen for the corresponding database table. In this context, deleting an additional table means removing the assignment to a data source table. To do this, place the cursor on the additional table and choose the appropriate function.

You can delete only if the fields of the table are not used in a query. That is always the case where fields of the table have not yet been included in any field groups of the InfoSet.

However, the system rejects any attempt to delete an additional table which has already been used in queries, as long as those queries still exist.

You can determine whether and in which queries a table field is used. To do this, place the cursor on a table field and choose *Environment* → *Directories* → *Queries for field* or choose *Queries for field* in the lower right-hand box. Change the queries such that the additional table fields are no longer used.

Creating Additional Fields

To create an additional field, proceed as follows:

1. Double-click on a data source and choose the function *Extras*.
2. Choose *Create*.

Another dialog box appears where you must first choose to define either an additional table, an additional field, an additional structure or code.

3. Choose *Additional field* and enter a field name. The field name must begin with a letter. The field name HEADER is reserved.

Additional Field SEATFREE

Long text: Free seats

Header: Free

Format

Type: C Length: 004 OutputLength: 010 Decimals:

LIKE Reference: SFLIGHT-SEATSOCC

Text identification

☐ Determine LIKE reference using text field

Text field:

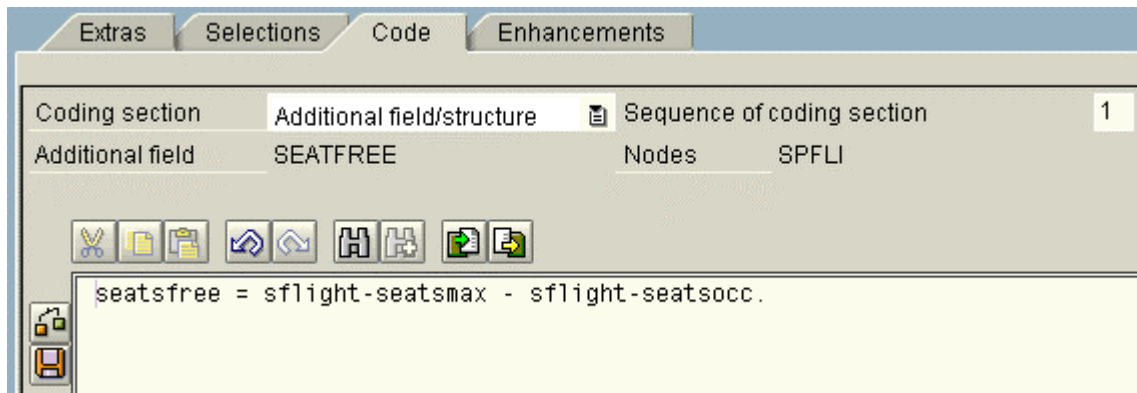
Sequence of code: 1 ☐ Code exists

✓ ✗

You can enter field definitions in the additional field maintenance screen. If required, you can also change the proposed sequence of the ABAP code for the additional field within the section of code for GET of the logical database table in the generated query report. This sequence could be important, for example, if you need the value of another additional field of the same logical database table for a calculation.

Finally, you must specify how you want to determine the value of the additional field. To maintain codes for this, choose *Codes for Extras* in the *Extras* tabstrip. This brings you to an Editor. All commands normally available in ABAP editors are also available here.

Creating Additional Fields



Note the editor function *Check*, which checks your code for syntax errors. Otherwise it can only be corrected when you use the functions *Check* and *Generate* for the InfoSet area as a whole. You should check carefully that the fields used in the code all contain valid values at the time the ABAP code is executed. The system helps you to do this by allowing the following fields:

- Parameters (see [Defining Parameters \[Seite 250\]](#))
- Data from the DATA code (see [Further Code \[Seite 257\]](#)).
- Fields of the data source table, to which the additional field is assigned and fields of superior logical database tables. In the case of the latter, any additional fields and additional table fields assigned there are also allowed.



If, for example, you have assigned the additional field to the SFLIGHT table of the logical database F1S, fields of the SPFLI and SFLIGHT tables are allowed. It is also possible to access the fields of additional tables as well as additional fields assigned to the table SPFLI.

- Other additional fields assigned to the same data source table.
In this case, you must ensure that the additional fields are processed in the correct sequence.
- Fields of a linked additional table assigned to the same data source table.
Here, you must ensure that the additional table is accessed before the additional field is processed.

Whenever you are on the screen for editing additional fields, you can change the following:

- the sequence of the ABAP code within the processing of data from the logical database table,
- description and header texts,
- format specifications, and
- Check box field *Generate text field using like-reference*. An attempt is made here to generate a text, used for reports, using the like-reference for the field.

However, you cannot change the assignment to a logical database table. To assign the additional table to a different data source table, you must delete the field and recreate it.

The example below shows the meaning of sequence numbers.

The screenshot shows the 'Additional Field SEATSFREE_PERCENTAGE' dialog box. It has several tabs: 'Long text', 'Header', 'Format', and 'Text identification'. The 'Long text' tab is active, showing the text 'Free seats in percentages'. The 'Header' tab shows 'Free'. The 'Format' tab shows 'Type P', 'Length 003', 'OutputLength 006', and 'Decimals 00'. The 'Text identification' tab is also visible, showing a checkbox for 'Determine LIKE reference using text field' and a 'Text field' input. At the bottom, there is a 'Sequence of code' section with a dropdown set to '2', a 'Code exists' checkbox, and a status bar with a green checkmark and a red X icon.

The value of the additional field SEATSFREE_PERCENTAGE is calculated with the predefined additional field SEATSFREE. This is why SEATSFREE_PERCENTAGE must have a higher number (2) than SEATSFREE (1), so that the correct value is in SEATSFREE when calculation takes place. In the ABAP Editor the calculation looks like this:

```
seatsfree_percentage = seatsfree / sflight-seatsmax * 100.
```

If you were to access fields of a linked additional table when performing calculations, the procedure would be similar.



To ensure that query reports can be generated without problems, it is important that it is clear which fields are accessed in every piece of code. If some code contains ABAP statements that access certain fields implicitly, then an ABAP FIELDS statement must also be used in the code to ensure that all fields used are explicitly named. You can find an example of this and further information in [Access Optimization in Queries \[Seite 342\]](#)

Assigning Additional Fields to a Field Group

Assigning Additional Fields to a Field Group

The defined additional fields are displayed on the screen for maintaining InfoSets at the end of the field list and can be assigned to a field group.



Do not forget to assign the defined additional fields to a field group. These fields can only be viewed by the end-user in the InfoSet and be used for defining queries after you have completed this assignment.

Displaying a List of Existing Additional Fields

To request a list of defined additional fields, select *Environment* → *Directories* → *Additional fields*. For each additional field, this list contains the data source table assignment and the specified sequence of the codes for processing data in the data source table. Subsequent lines contain the description and the ABAP code.

Deleting Additional Fields

Deleting Additional Fields

If you want to delete an additional field, you must first call the additional information screen for the corresponding database table. To do this, place the cursor on the additional field and choose *Delete*.

This is possible only if the additional field is not used by any queries. If you want to delete an additional field, even if it is used in a query, you must first change all the queries affected. To do this, follow the steps described in [Deleting the Assignment of Additional Tables \[Seite 240\]](#).

Creating Additional Structures

To create an additional structure, proceed as follows:

3. Double-click on a data source and choose the function *Extras*.
4. Choose *Create*.

Another dialog box appears where you must first choose to define either an additional table, an additional field, an additional structure or code.

5. Choose *Additional Structure*. Enter the name of the additional structure and then the name of a structure from the ABAP Dictionary.

Additional structures are actually additional fields with a non-scalar type of data. This data type must always be a (flat) structure from the ABAP Dictionary.

An additional structure must be assigned to a node of the data source in the same way as an additional field. However, you can find all individual fields from the additional structure in this node of the data source. So, additional structures offer a simple way of calculating more than one additional field at the same time.

Assigning Structure Fields to Field Groups

Assigning Structure Fields to Field Groups

The fields for the inserted structure are displayed at the end of the fields list.

The screenshot shows the SAP Query Designer interface. On the left, the 'Data fields' list is expanded, showing a hierarchy of fields. The 'Flight demo table' (SFLIGHT) is highlighted. On the right, the 'Additional information on node SFLIGHT' table is displayed, showing the structure and meaning of the fields.

Name	Art	Bedeutung
1 PERSDATA	Add. structure	Personnel data
1 SEATSFREE	Addit. field	Seats free

They can be assigned to a field group using the method already described.



Do not forget to assign the relevant fields from a linked structure to a field group. These fields can only be viewed by the end-user in the InfoSet and be used for defining queries after you have completed this assignment.

Deleting the Assignment of Additional Structures

If you want to delete an additional structure, you must first call the additional information screen for the corresponding database table. In this context, deleting an additional structure means removing the assignment to a data source table. To do this, place the cursor on the structure and choose *Delete*.

You can only delete if the fields of the structure are not used in a Query. That is always the case where fields of the structure have not yet been included in any field groups of the InfoSet.

However, the system rejects any attempt to delete a structure that has already been used in Queries, as long as those Queries still exist.

You can determine whether and in which queries a table field is used. To do this, place the cursor on a structure field and choose *Environment* → *Directories* → *Queries for field* or choose *Queries for field* in the lower right-hand box. Change the Queries so that the additional structure fields are no longer used.

Creating Extra Selection Fields

All queries have a selection screen that is called when a query is run. With the entries he or she makes on this selection screen, the user determines which data should be read from the dataset. As described in [Executing Queries Online \[Seite 103\]](#) selection screens are divided into several parts. In the case of InfoSets associated with logical databases, part of the selection possibilities on the screen are determined by the logical database. Another portion of what appears on the selection screen can be determined when defining the query (see [Extending the Selection Criteria \[Seite 149\]](#)) or is automatically retrieved for you. In addition to these two possibilities, you can also define additional selection fields (parameters and selection criteria) in an InfoSet. These selection fields appear on the selection screens of all queries created using the InfoSet and form a set of standard selection fields for these queries.

When using InfoSets associated with logical databases, you should check to see if the logical database's dynamic selections can be used for selections of this sort, seeing as they are a much more efficient way of selecting data. When using InfoSets without an underlying database (see *Creating and Changing InfoSets without an Underlying Database*), selection fields defined for a InfoSets are the only way to provide standard selections.

To define or change selection fields for a particular InfoSet, call up *Selections* on the functional group creation screen. A window is displayed containing all selection fields for the InfoSet.

This window contains the sequence number, the names of the selections and a checkbox respectively, in order to determine whether the selection is to appear on the selection screen of the InfoSet query.

In the case of InfoSets associated with logical databases, all selections for that logical database are also displayed. You cannot change these selections.

The window's first column contains a number. This number determines the order in which the code for each individual selection field will appear on the selection screen. This number only affects those selection fields that have been defined using the InfoSet. These are displayed on the selection screen after selection fields from the logical database and before all query specific selection fields.



If two selection fields have the same sequence number, then the order they appear in on the selection screen is uncertain.

This window also allows you to *Create*, *Change* or *Delete* selection fields. When using *Change* and *Delete*, you must place your cursor on the appropriate selection field entry. You can also trigger the *Change* function with a double-click.

For every selection that is defined by the InfoSet or provided by the logical database, you can create code at the AT SELECTION-SCREEN in order to carry out checks. For this, the cursor must be placed on a selection, and then the function *Check Code for Element* should be called.

The code at AT SELECTION-SCREEN has to be made separately for each selection. When generating a query report, you ensure that the check coding for all selections is grouped together at a joint time AT SELECTION-SCREEN.

Parameters which are defined for an InfoSet belong to every query in the InfoSet. They are, however, not fields which you can assign to a logical database table. For this reason, they cannot appear as fields in a query list.

You can use parameters in InfoSets wherever ABAP code occurs, that is in any of the following situations:

- when specifying values for the key fields of linked additional tables
- the code of additional fields

- the code for the events GET/GET LATE
- the code for the events START/END

Parameters are not associated with particular logical database tables.

As with parameters, selection criteria defined for an InfoSet belong to every query of that InfoSet and appear on the selection screen.

You can use selection criteria wherever ABAP code occurs, for example, in any of the following:

- the code of additional fields
- the code for the events GET/GET LATE
- the code for the events START/END



Selection fields must always be backed up by ABAP code to ensure that they function properly.

Further information can be found in the following explanations of parameters and selection criteria.

Creating Parameters

Creating Parameters

To create a parameter, proceed as follows:

1. Choose *Selections* in the InfoSet maintenance screen.
2. Choose *Create*.
3. In the dialog box that appears, determine whether you are defining a parameter or a selection criterion and enter a name for the selection containing up to eight characters.
4. Define the parameter.

Parameter DEPART

Sequence on selection screen 1

Definition

Description Departure airport

Selection text Departure airport

Format Type C Length 003

LIKE SPFLI - AIRPFROM

Extras OBLIGATORY

Error message

Check code

By specifying a sequence on the selection screen, you can determine the arrangement of the parameters. This is generally in ascending order. If two parameters have the same sequence specification, the order in which they appear on the selection screen is uncertain.



The sequence number applies equally to parameters and selection criteria.

The text you enter on the *Description* line serves as documentation and should describe the selection criterion as well as possible. It should describe the parameter. The *Selection text* appears on the selection screen of every query report within that InfoSet. If you do not enter a selection text, the system uses the description as the selection text.

When defining parameters, you have the same options as when you define additional fields.

If you want to assign attributes such as type and length specifications to the parameter, use the field *Additns*. There, you can specify all the additions allowed with the ABAP statement `PARAMETERS`. If required, you can use several additions, one after the other.

The system then performs a syntax check.

For every selection that is defined by the InfoSet or provided by the logical database, you can create code at the AT SELECTION-SCREEN in order to carry out checks. Ensure that the cursor is positioned on a selection in the overview screen for the available selections, and then call the *Check code for element* function.

The code at AT SELECTION-SCREEN has to be made separately for each selection. When you generate a query report, checks are made to ensure the check code for all selections is collected at the same AT SELECTION SCREEN and that only those selections that appear on the selection screen are referred to.



You should also evaluate parameters that you define. A typical example would be CHECK <parameter> to evaluate parameters in a GET event, in order to allow only certain table lines to be processed.

Creating Selection Criteria

Selection criteria are created in much the same way as parameters (see [Creating Parameters \[Seite 252\].](#))

Selection Criterion DATE

Sequence on selection screen 1

Definition

Description Flight date

Selection text Flight date

Format FOR SPFLI

Extras OBLIGATORY

Error message

Check code

By specifying the sequence, you can control the arrangement of selection criteria on the selection screen. This is always in ascending order.



The sequence number applies equally to parameters and selection criteria.

The text you enter on the *Description* line when defining a selection criteria serves as documentation and should describe the selection criterion as well as possible. The selection text is displayed on the selection screen of every query report for the InfoSet. If you do not enter a selection text, the system uses the description as the selection text.

Selection criteria assignment for a data source node takes place using the FOR field that has to be assigned to each selection criterion. The selection criterion is set on the query selection screen if the corresponding data source node is in any way relevant to the query.

If you want to assign special attributes to the selection criterion, use the *Additns* line. There, you can specify all the additions allowed with the ABAP statement SELECT-OPTIONS. If required, you can use several additions, one after the other.

The system then performs a syntax check.

For every selection that is defined by the InfoSet or provided by the logical database, you can create code at the AT SELECTION-SCREEN in order to carry out checks. Ensure that the cursor is positioned on a selection in the overview screen for the available selections, and then call the *Check code for element* function.

The code at AT SELECTION-SCREEN has to be made separately for each selection. When you generate a query report, checks are made to ensure the check code for all selections is collected at the same AT SELECTION SCREEN and that only those selections that appear on the selection screen are referred to.

Displaying a List of Existing Parameters

Displaying a List of Existing Parameters

To display a list of existing parameters for an InfoSet, select *Environment* → *Directories* → *Selections*. You then see the defined parameters and selection criteria, the sequence in which they appear on the query selection screen, the descriptions, the selection texts and the type definition specifications.

Further Code

Previous sections described how to use additional tables, additional structures and additional fields to extend the scope of a query for evaluating data from a data source when you create an InfoSet. Additional tables, additional structures and additional fields have two common features:

- The point (event) at which SAP Query reads an additional table or performs a calculation on an additional field depends on the logical database table, since the associated ABAP code becomes part of the GET event processing that table.
- Any query for the InfoSet can address both additional fields and additional table fields, provided they are assigned to a field group of that InfoSet.

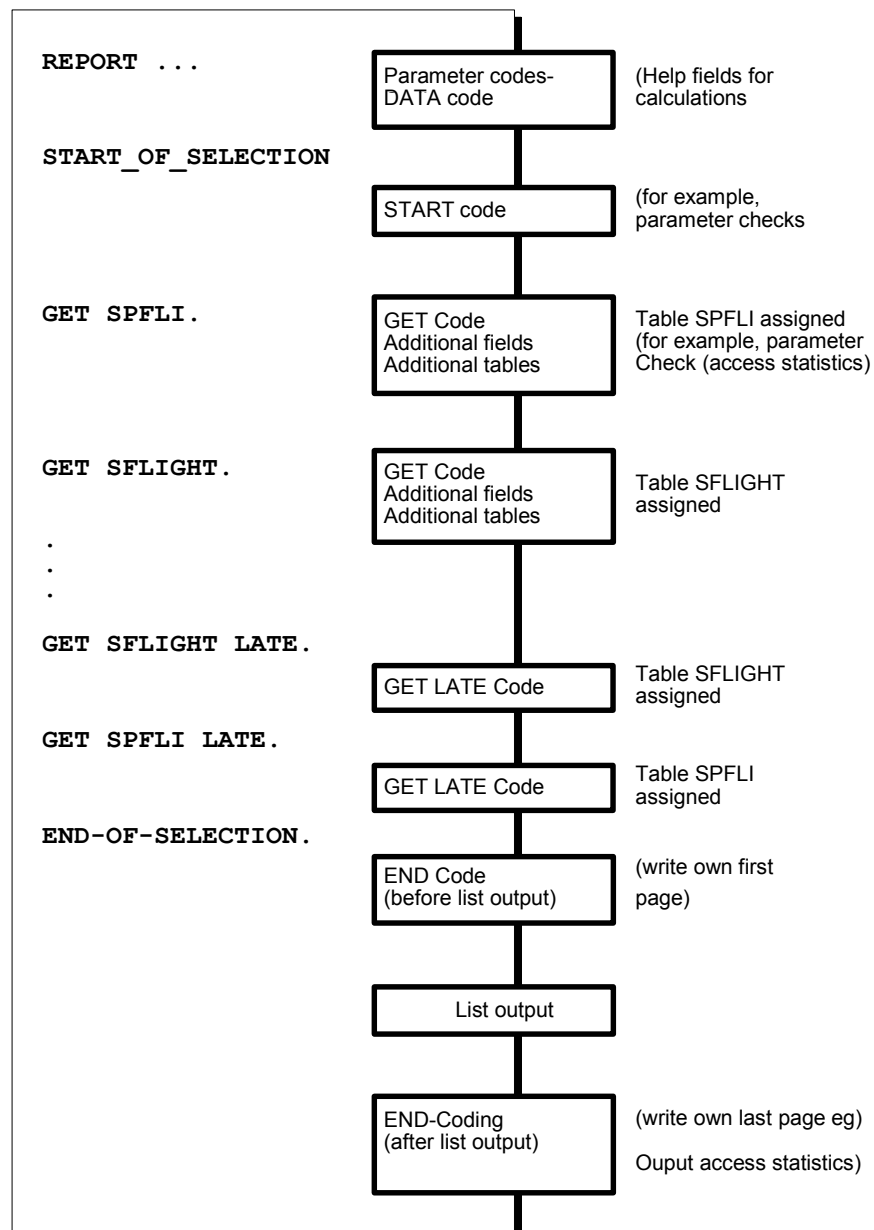
The previous sections also discussed parameters and selection criteria which you use to create selection options.

This topic discusses additional options which make additional tables, additional structures, additional fields, parameters and selection criteria more flexible and give you more room to move, but cannot be addressed in queries for the InfoSet. These options generate ABAP code for all queries of an InfoSet, but this is transparent to the end-user.

- **DATA Code**
This code declares data to be used in a report (for example, variables for calculations).
- **START Code**
This code appears under the event START-OF-SELECTION which precedes the first database access.
- **GET code**
This code appears under the event GET for a logical database table, but does not depend on a particular additional field.
- **GET LATE code**
This code appears under the event GET LATE for a logical database table.
- **END Code**
This code appears under the event END-OF-SELECTION either before or after output of the list.
- **TOP-OF-PAGE code**
This code is copied into every generated report and appears at TOP-OF-PAGE just before the instructions for output of the page header.

The diagram below shows how the structure of sections of code you can amend in a query report for the InfoSet FLBX (using the logical database F1S).

Further Code



To request a list of defined codes of the InfoSet, select *Environment* → *Directories* → *Codes*.
The following sections describe the different sections of code in more detail.

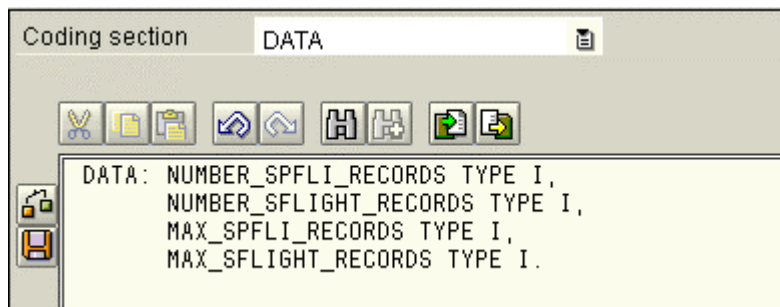
DATA Code

The DATA section allows you to declare data used in other sections of code. In query reports, this section appears before all other event key words, so that the declared data is known to, and available in, all other sections of code.

You use the DATA section mainly to declare variables (e.g. number variables, counters, additional field strings, field symbols) which are required for performing calculations in other sections of code or specifying values for key fields of a linked additional table.

You are strongly recommended to declare all such variables in the DATA section rather than "hiding" them in any other sections. Otherwise, resolving syntax check errors becomes an unnecessarily complicated procedure. The DATA section is transparent to the end-user.

To create or maintain DATA code from the InfoSet maintenance screen, choose the symbol *Code*. With the aid of F4 help, you can then choose the code type. An editor is called where you see an existing DATA section or you can create a new one.



You can choose *Check* to check your code for syntax errors.

When declaring data, you can use LIKE to refer to any objects defined in the ABAP Dictionary. You cannot refer to additional fields.

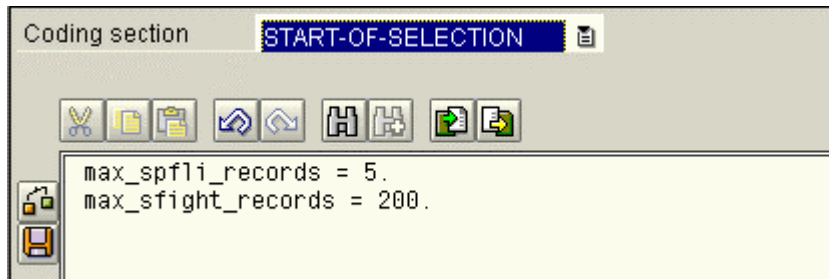
If you want to delete DATA code, you can do this from within the editor for DATA definition.

START Code

START Code

The code for the START-OF-SELECTION event in a report is executed before starting any database accesses, meaning before the first data record of the highest table in the logical database hierarchy is read. The code's purpose is to initialize all the variables used when reading data from the database. Particularly additional fields or help variables that are used for filling key fields from connected additional tables can also be initialized here. Selection screen entries can also be checked. The START code is transparent to the end-user.

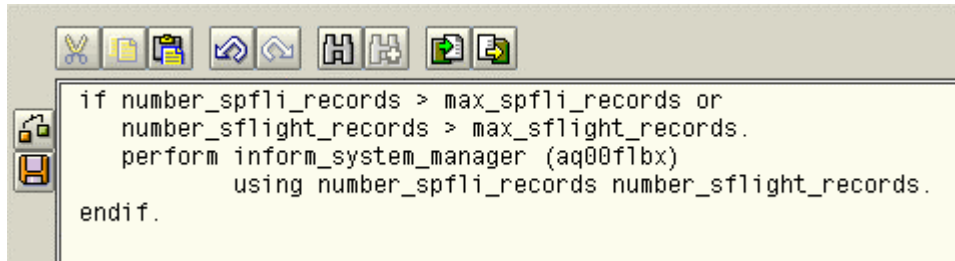
Choose the pushbutton *Code* in the InfoSet maintenance screen and the *START Code* field help to create or maintain START Code.



Choose *Check* to check your code for syntax errors.

END Code

The END-OF-SELECTION code consists of two parts (*Before list output* and *After list output*). The first part is processed before the list is output and the second part afterwards. Choose the pushbutton *Code* and, with the aid of F4 help, END-OF-SELECTION (before list output) or (after list output).



This division means that you can make your own additions to the list created by the query both before and after the list is output. Since your own additions are always output on a separate page, you can use this to generate a first and a last page that are output independently of the query definition. You should, however, note that these special additions to the output cannot be displayed by the layout display in the query maintenance component.

The difference between the code for START-OF-SELECTION and that for END-OF-SELECTION (before the list is output) is that whilst the START-OF-SELECTION code is processed before the first database access, the END-OF-SELECTION code (before the list is output) is processed after the last database access, but before the list is output.

If a first page is to be generated via code, this should not be done in the START-OF-SELECTION code. In this case the page initially would be created when the query is executed, but it would not be created in interactive functions that rebuild the list (printing, reduction and compression of basic lists). If the first page is created in END-OF-SELECTION code (before the list is output), this error does not occur.

Any ABAP statements may be used in the code. Any data objects may be accessed. However, it is better to restrict to the following objects:

- Parameters from the InfoSet and the logical database
- Selection criteria for the InfoSet and the logical database
- global data from the DATA code

Fields in logical database tables or additional tables and additional fields do not have a defined value at this point.

GET/GET LATE Code

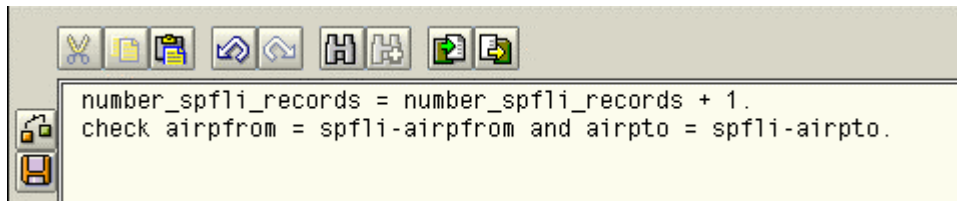
GET/GET LATE Code

If you want to perform calculations or other operations which are not associated with certain additional fields, you can formulate ABAP code for the GET and GET LATE events of a logical database table when creating an InfoSet. The code for GET events is particularly important if you have defined parameters or selection criteria for the InfoSet. It is here that you must define the code to check whether a table line is to be evaluated or not.

As far as linking to logical database tables is concerned, ABAP code for the GET and GET LATE events is treated in each case like an additional field. The code becomes part of every query report generated for an InfoSet. Since that code is not assigned to any InfoSet, it is transparent to the end-user.

To create or maintain GET/GET LATE code for a logical database table from the InfoSet maintenance screen, choose the symbol *Code*, and *GET*, or *GET LATE*, with the F4 help.

If you chose one of these functions then first enter in an entry field, at the top right of the screen, a number for the sequence.



The sequence is only relevant for the code for the GET event. This code appears together with the code for additional fields and SELECT statements for linked additional tables in the GET event for the logical database table concerned.

Usually, the GET event code is used for initialization and precedes additional fields and additional tables in the sequence.



To ensure that query reports can be generated without problems, it is important that it is clear which fields are accessed in every piece of code. If some code contains ABAP statements that access certain fields implicitly, then an ABAP `FIELDS` statement must also be used in the code to ensure that all fields used are explicitly named. You can find an example of this and further information in [Access Optimization in Queries \[Seite 342\]](#)



If you want to assign the code to a different logical database table, you must delete the code and then redefine it.

TOP-OF-PAGE Code

The TOP-OF-PAGE code is copied into every generated report and appears at TOP-OF-PAGE just before the instructions for output of the page header. If this code includes WRITE statements, then the output will appear in the list before the page header that is created by the query. If the list is printed with a standard page header, then the output appears between the standard page header and the page header created by the query. You should note that these additions to the output cannot be displayed by the layout display in the query maintenance component.



When outputting using the SAP List Viewer, the code at TOP-OF-PAGE has no significance.

The TOP-OF-PAGE code is maintained by selecting the symbol *Code*. Choose TOP-OF-PAGE using the F4 help. Any ABAP statements may be used in the code. Any data objects may be accessed. However, it is better to restrict to the following objects:

- Parameters from the InfoSet and the logical database
- Selection criteria for the InfoSet and the logical database
- global data from the DATA code

Fields in logical database tables or additional tables and additional fields do not have a defined value at this point.

The %HEAD variable can be used to specify which page header should be output. This variable is available in the standard settings and takes one of the following values:

AAA	Output at END-OF-SELECTION (before the list is output)
GGG	Output the basic list
Txx	Output the statistics xx (xx=01-99)
Rxx	Output the ranked list xx (xx=01-99)
WWW	Output the conversion table
ZZZ	Output at END-OF-SELECTION (after the list is output)

The value of the %HEAD variable may be queried but may not be changed, since it is an internal runtime system variable.

The use of TOP-OF-PAGE code allows the standard page header to be extended. However, it is important to note that if this is done then the changes will take effect on all queries that work with this functional area.

Application-specific Enhancements

SAP Query is a generic tool that can be used with all applications within SAP systems for reporting tasks. This generic approach has, until now, prevented SAP Query from utilizing specific application knowledge (specific application logic).

The application HR is, however, an exception. There is a range of specific HR solutions integrated into SAP Query (InfoSet generation, query target groups, and so on). These solutions are strictly coded, meaning that changes or enhancements are only possible if you modify SAP Query.

From the current Release, work has begun on enhancing SAP Query so that it is possible to evaluate application-specific logic without having to modify SAP Query itself. The procedure for setting up these enhancements is similar to that for defining and implementing [Business Add-Ins \[Extern\]](#).

Specific interface methods are called at different times, for example, when creating an InfoSet or during a query runtime. These interface methods are either implemented by a basis class (standard) or from a special class containing additional application logic. If you want your queries to use application-specific functionality rather than basis functionality for a particular service using an InfoSet, then you can deposit the class that implemented the corresponding interface in the InfoSet. The InfoSet then becomes a carrier for all the information on application-specific logic.

A practical example should help explain the procedure:


The interface methods `IDENTIFY_TEXT` or `READ_TEXT` for the `IF_TEXT_IDENTIFIER` interface are called for automatic text identification when creating an InfoSet and for reading relevant texts on query runtimes. This interface is implemented by the basis class `CL_TEXT_IDENTIFIER` and this implementation in turn is used by default. Texts that cannot be identified by the basis class could possibly be determined by additional logic in a particular application context. For example, the `CL_HR_TEXT_IDENTIFIER` class in the HR environment. In principle, the application classes can either re-implement the `IF_TEXT_IDENTIFIER` interface or be derived from the basis class. The relevant class for text identification is specified in the InfoSet; corresponding implementation for the interface method is called for reading texts in all queries for this InfoSet.

You can use the program `RS_TEXT_IDENTIFY_TEXT` to test text identification. Essentially, the class `CL_TEXT_IDENTIFIER` evaluates the ABAP dictionary. It can happen that texts exist for particular fields, but the link between value field and text field is not apparent in the ABAP dictionary. If this occurs, you can define exceptions in SAP reference IMG (*Basis* → *SAP Query*): You can, for example, specify a function module for creating a text, or create a text for a field in a table directly. The procedure for defining exceptions is described in the documentation for the `IF_TEXT_IDENTIFIER` interface.

The following interfaces are defined for Release 4.6C:

Automatic text identification	Interface: <code>IF_TEXT_IDENTIFIER</code> Methods: <code>IDENTIFY_TEXT</code> , <code>READ_TEXT</code> Basis class: <code>CL_TEXT_IDENTIFIER</code>
InfoSet generator (generation and preassignment of field groups)	Interface: <code>IF_QUERY_INFOSET_GENERATOR</code> Methods: <code>MODIFY_INFOSET</code> Basis class: <code>CL_QUERY_DATASOURCE_GENERATOR</code>

You can enter a special class in the dialog box when creating an InfoSet (*More Options*).

The specified service classes are displayed in InfoSet maintenance in the *Application-specific Enhancements* tabstrip ()

InfoSets in the HR Application

You can use SAP Query in HR to report on HR data. Queries are maintained as described in [Creating Queries \[Seite 136\]](#). The special features of queries created for HR are described in [Maintaining Queries in the Human Resources Application \[Seite 195\]](#). The maintenance procedure for HR InfoSets differs from the described procedure inasmuch as HR data fields are grouped together in infotypes.

InfoSet management in SAP Query is also used for InfoSet Query. For further information, see [Functions for Managing InfoSets \[Seite 201\]](#).

If you want to create InfoSets for HR, you can use logical databases PNP, PAP, and PCH (see [HR Logical Databases \[Seite 270\]](#)). The database you must use to create your InfoSet depends on the component in which the data you want to report on is stored.


The reports you can execute using InfoSets based on logical databases PNP or PCH are similar, but differ in that they can select different objects. The following table describes the connection between the logical database, and the infotypes you can include in an InfoSet. It also provides you with one or two examples of reports that you can execute using the appropriate InfoSets.

Logical database	PNP	PCH	PAP
Selection of	Persons	Objects from Personnel Planning	Applicants
Infotypes that can be included in the InfoSet	Infotypes for <ul style="list-style-type: none"> Personnel Administration (0000-0999) Time Management (2000-2999) Payroll infotypes Infotypes for Personnel Planning objects that can be related to persons 	If the object type is specified: <ul style="list-style-type: none"> Infotypes for the object type Infotypes for objects that can be related to the specified object type If the object type is not specified: <ul style="list-style-type: none"> All infotypes 	<ul style="list-style-type: none"> Infotypes for Recruitment (4000-4999) Some infotypes for Personnel Administration (such as 0001 and 0002)
	<ul style="list-style-type: none"> Customer infotypes 		

Reporting examples	<ul style="list-style-type: none"> • Selection of all persons who participated in a specific business event, output of prices for reserved business events • Selection of all persons assigned to a specific personnel area, output of qualifications held by these persons 	<ul style="list-style-type: none"> • Selection of all business events held in London in March, output of all persons who participated in these business events • Selection of all positions assigned to a specific organizational unit, output of all persons assigned to the positions 	<ul style="list-style-type: none"> • Selection of all applicants hired last year to work on special projects, output of addresses for the applicants selected
---------------------------	---	---	--

Creating InfoSets

The maintenance procedure for HR InfoSets differs from the procedure described so far in this section inasmuch as HR data fields are grouped together in infotypes. To set up an InfoSet for the HR application, proceed as follows:

1. On the initial screen for maintaining InfoSets, enter a name for the InfoSet and choose .
2. On the next screen, enter a name for the InfoSet and select one of the HR logical databases in accordance with your reporting requirements.



Customer infotypes can be created on all HR logical databases. In each individual case, therefore, you must decide which database to select so that you can report on customer infotypes.

This screen enables you to enter an [authorization group \[Extern\]](#). All of the queries that are subsequently created using this InfoSet can only be executed by persons who have this authorization group.

3. Choose .

This takes you to the *Infotype Selection for InfoSet <InfoSet name>* dialog box. It contains all of the infotypes that you can access using the selected logical database.

- If you use **logical database PCH** to create an InfoSet with which to select objects in InfoSet Query, select the object type first.

Once you have selected the object type, you can select the object type's infotypes. Furthermore, all of the object types that can be related to the selected object type are listed below *Infotypes of related objects*. The next level in this tree outputs all of the

InfoSets in the HR Application

relationships that can exist between the object type in question and the object type that can be selected. All of the selected object type's infotypes are displayed on the last level.

- If you use **logical database PNP** to create an InfoSet, you can use the infotypes from Personnel Administration. They are grouped together according to the current user group in Personnel Administration.

Furthermore, all of the object types that can be related to the *persons* object are listed below *Infotypes of related objects*. The next level in this tree outputs all of the relationships that can exist between the respective object type and the *person* object type. The following relationships, for example, can exist between persons and qualifications: *fulfils*, *has potential for*, *interests and preferences*, and *dislikes*. All of the selected object type's infotypes are displayed on the last level.

- If you use **logical database PAP** to create an InfoSet, you can use the infotypes from Recruitment and some infotypes from Personnel Administration.

4. Choose the infotypes that are required in the InfoSet.

A field group is created in the InfoSet for each infotype that you select. The name of the field groups corresponds to the name of the infotype or consists of the object name, relationship name, and infotype name (for example, *qualification/fulfils/object*).

5. Choose .

This takes you to the *Change InfoSet <InfoSet name>* screen. You now have the option of creating field groups and assigning fields as required for non-HR InfoSets. Field groups that correspond to infotypes and already contain fields, however, are always created for HR InfoSets. The field groups are displayed in an overview tree in the top right section of the screen.

The infotypes that you included in the InfoSet are displayed in an overview tree on the left of the screen. The infotype fields that are already included in field groups are displayed in a different color, and the field group ID is displayed.



In the standard system, a field group is created automatically for each infotype that you included in the InfoSet (a field group corresponds to an infotype).

In the standard system, each field group contains the infotype-specific fields. To ensure that working with the InfoSet is as easy as possible, you are advised to restrict your use of fields in each field group to those you really require. This means you should remove fields that are not required.

An infotype's fields must only be assigned to the pertinent field group. Make sure this assignment is correct. If the assignment is incorrect, the InfoSet could be rendered unusable.

When an InfoSet is created, the following fields are transferred automatically to the first field group:

- Logical database PNP *Personnel number*
- Logical database PAP *Applicant number*
- Logical database PCH *Object ID, plan version, and object type*

6. Determine the fields that must be included in the field groups of your InfoSet. If you require further information, see [Assigning Fields to a Field Group \[Seite 228\]](#).



If you want, you can change the default sequence of field groups and fields as required using drag & drop.

7. To save the InfoSet, choose
8. To generate the InfoSet, choose



On the *Change InfoSet (InfoSet name)* screen, you can choose *Edit → Change infotype selection* to add more infotypes to the InfoSet, or to remove infotypes from the InfoSet. Remember to regenerate the InfoSet afterwards.

This screen also enables you to update InfoSets if, for example, the system contains new additional fields for specific key values. To do so, choose *InfoSet → Additional functions → Update additional HR fields*.

9. Go back to the initial screen for InfoSet maintenance.
10. Choose *User group assignment*.
11. Select a user group, and save your entry.

HR Logical Databases

In Human Resources (HR), the following logical databases can be used as a data source for HR InfoSets:

- PNP
- PAP
- PCH

By selecting a logical database, you determine the HR data that can be reported on using an InfoSet.

Logical Database PCH

This logical database generally enables you to report on all HR infotypes. However, you are advised not to use this logical database unless you want to report on Personnel Planning data.

Logical Database PNP

Use logical database PNP to report on HR master data. It is possible to use logical database PCH to access this data, but PNP meets such reporting requirements more quickly because it is best suited to the task of selecting persons.

Logical database PNP enables you to access HR master data and infotypes from Personnel Planning. For example, you have the following options:

- Reporting on the costs, number of attendees booked, and instructor for a business event on which an employee is booked
- Reporting on working time and planned compensation for a position that an employee occupies
- Reporting on the validity and proficiency of a qualification that an employee fulfils

From a technical perspective, this means you can use PNP to report on all of the infotypes that exist for objects (infotype 1000) that have a direct relationship (infotype 1001) with the *Person* object.



The ability to access infotypes from Personnel Planning using logical database PNP is a special feature that you can only use in the context of SAP Query and Ad Hoc Query. You cannot use this functionality for ABAP reports you programmed yourself.

You can also use logical database PNP to report on data from Personnel Time Management (infotypes 2000 to 2999) and Payroll (special payroll infotypes for the USA and customer infotypes; for more information, access Customizing for the *Human Resources Information System* and see *Payroll Results*).

Logical Database PAP

Logical database PAP enables you to access data from Recruitment.

See also:

[HR InfoSets for InfoSet Query \[Seite 70\]](#)

[InfoSets in the HR Application \[Seite 266\]](#)

Creating InfoSets without an Underlying Logical Database

As well as producing reports on logical databases, you can also use SAP Query for reports whose data did not come from a logical database.

There are four ways of executing reports without any underlying database: You can evaluate sequential datasets.

- You can execute a report with the help of a table join. The result set of a table join is a table, each of the lines of which contains all the fields of all the tables used in the join.
- You can read the contents of tables directly. Here, the system uses a SELECT statement to retrieve data.
- You can evaluate any dataset. To do this, you must write a data retrieval program yourself and store it as a model program.
- In this case, the system retrieves data as it would for a report using a logical database.

Select one of the four data retrieval methods specified.

These evaluations use largely the same methods as those described in [Creating and Changing InfoSets \[Seite 218\]](#).

If the InfoSet is based on a table join, then you should specify the name of the first table of the join in a table join.

If you opt for data retrieval using a program, you must specify the name of the data retrieval program.

If you decide to read a sequential dataset, you must specify a file name. The selection button *Sequential dataset* is shown when you choose the pushbutton *Further options*.

With the last two options, also enter the name of a structure listed in the ABAP Dictionary. This structure must reflect the record structure of the dataset you want to evaluate.



The structure must be defined before you can use it. For further information about creating structures in the ABAP Dictionary, refer to the ABAP Dictionary documentation.

Graphical Table Join Definition

If several tables are all linked within a SELECT statement, this is known as a join. The result set is a table, each of the lines of which contains all the fields of all the tables used in the join. The links between the various tables used in the join are all specified separately. These specify the exact combinations of records from the individual tables that become part of the result set.



The result of a table join is also a (flat) table! Therefore hierarchical relationships between tables cannot be analyzed using a table join. This requires logical databases.

For more detailed information about table joins you should refer to the online documentation for the ABAP SELECT statement.

You can also define table joins in a graphic design mode if certain technical software requirements have been fulfilled. The system then simulates the join including the relationships between its individual tables.

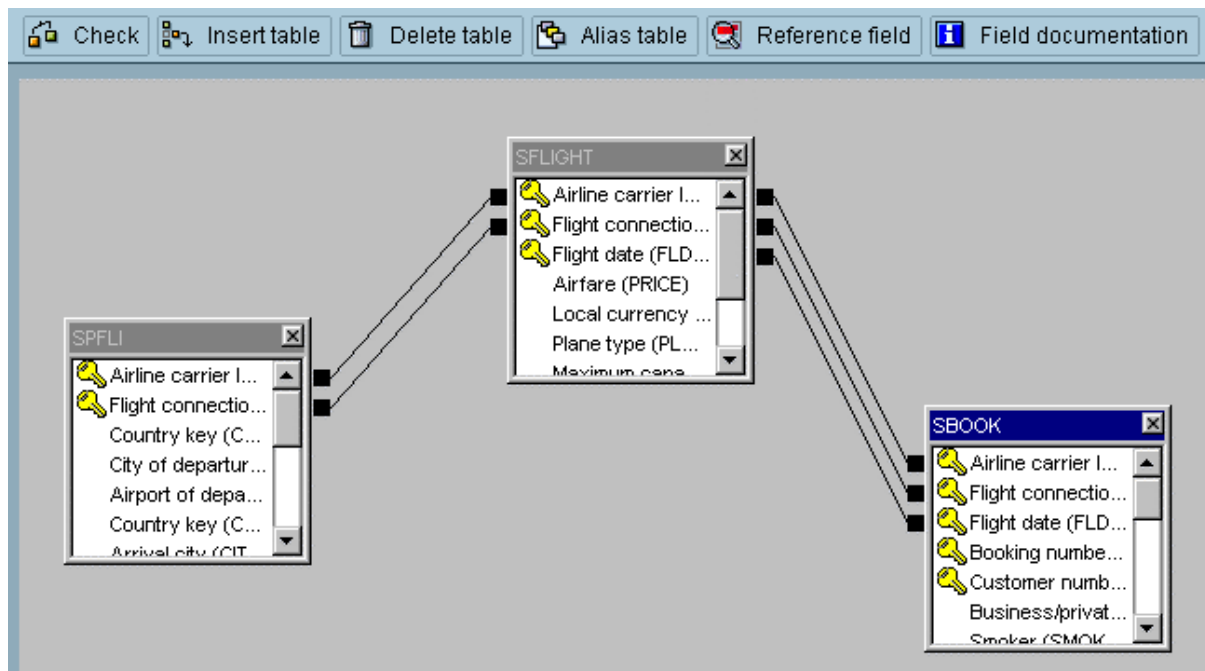


Table joins must already be defined before you maintain InfoSets and QuickViews. With InfoSets, you can switch off graphical join definition under *Settings* → *Settings...* and create joins using the procedures described in [Classical Table Join Definition \[Seite 274\]](#) instead.



If certain technical software requirements are fulfilled, graphical join definition always appears as the standard setting.

The following functions are available on the *Join Definition* screen (Note: Those functions available in pushbutton form are written in *italics*):

Graphical Table Join Definition

<i>Insert table</i>	Enter the name of the table you want to insert in the dialog box. The table is then displayed in a window, that is the table fields and their long texts are displayed. Key fields are marked with a relevant icon. The dialog box can be moved if you so wish.
<i>Propose join conditions</i>	The system inserts standard proposals for joining two tables. The defaults suggested are derived based on the foreign key dependencies stored in the Dictionary or on the key fields of the tables used in the join.
Create links	The join conditions for your tables are displayed in line form. In order to create such a line, click on a starting field. Then, leaving your left mouse button depressed, drag your cursor to the field that you want to link to. If linking of the two fields is allowed, the system creates a line after you have released the left mouse button.
Define link type	Place the cursor on the line and depress the right mouse button. A context menu appears where you can choose to display or delete join conditions. If you choose to display the join conditions, a dialog box appears where you can set the join type to either inner join or left outer join. You can find further information Classical Table Join Definition [Seite 274] .
<i>Delete table</i>	A dialog box appears where you can enter the name of the table you want to delete. Alternatively, you can select the corresponding pushbutton in the table window (<i>Closebox</i>).
<i>Alias table</i>	Choose <i>Alias table</i> if you want to include the same table in a join twice. Define an alias name for the appropriate table as described in Assigning Additional Tables [Seite 234] . You can include the table in your join under both its original name and its alias.
<i>Check</i>	The system checks to see if the links between tables make sense, or if tables have been inserted, but not joined.
<i>Reference field</i>	Placing your cursor on a quantity or currency field and choosing <i>Reference field</i> causes the system to display the reference table containing that field's corresponding unit of measure or currency.
<i>Field documentation</i>	Place your cursor on a field for which you wish to display documentation. Information about the field's technical definition is displayed. Two fields can be linked only if both fields have the same data type in the Dictionary (including the length attribute). This means that two fields can be linked if they have the same domain.

You can find out about further restrictions and rules on how to use joins in the section on [Classical Table Join Definition \[Seite 274\]](#).

When you are done defining a table join for a QuickView, choose *Back* from the *Join Definition* screen to continue maintaining QuickViews. Your join definition is then saved together with your QuickView.

If you are using the graphical join definition while maintaining an InfoSet, choose *Back* to return to InfoSet maintenance and save manually.

Definition of a Table-Join without Graphics

Definition of a Table-Join without Graphics

If several tables are all linked within a SELECT statement, this is known as a join. The result set is a table, each of the lines of which contains all the fields of all the tables used in the join. The links between the various tables used in the join are all specified separately. These specify the exact combinations of records from the individual tables that become part of the result set.



The result of a table join is also a (flat) table! Therefore hierarchical relationships between tables cannot be analyzed using a table join. This requires logical databases.

More detailed information about table joins you should refer to the online documentation for the ABAP SELECT statement.

If you want to allow the user of the query to run reports, you should create an InfoSet for which you enter the name of the first table in the join in the *Table* field on the *Title, Database* screen and select the *Table join* field.

The table must be entered in the ABAP Dictionary and must exist in the database.

For technical reasons the first table in a join cannot be changed later. Choosing *Continue* takes you to the *Tables in the join* screen. Other information needed for the table join can be entered here.

All the tables to be included in the join are entered on the left-hand side of this screen. The type of link to be used between each pair of tables must be specified:

- inner join (default setting):
A record is included in the result set for each record in the first table for which a record in the second table exists satisfying the link conditions (see below).

Definition of a Table-Join without Graphics

- left outer join

Each record of the first table is included in the result set. If there is no record in the second table satisfying the link conditions for a record in the first table, then a record containing fields with initial values is used for the second table.



If you want to include a table in a join more than once, you have to use an alias table. Define an alias name for the appropriate table as described in *Assigning Additional Tables*. You can then include the table in your join under both its original name and its alias.

Link conditions must be defined between each pair of tables in the join. Two steps are needed to do this. If just two tables are being linked in a join, then the first step is simply to call the *Define condition* function. The two tables will then appear on the right hand side of the screen in a list of table pairs. If more than two tables are being linked together in a join, then two tables should be selected before the *Define condition* function is called. Then when the function is called these two selected tables will appear on the right hand side of the screen in the list of table pairs.

The *Define condition* function simply determines the tables between which link conditions are to be defined. The second step is to specify each individual condition.

A link condition is specified by calling the *Specify condition* function. This function can be called for each table pair using the pushbutton on the screen to the right of the table pair. When the function is called another screen appears in which the link conditions for this table pair are specified. Although any conditions are possible in a table join, the query supports only the case that two fields from the two tables each have the same value (equality relationship).

The first time that the *Specify condition* function is called for a table pair, defaults can be defined for the condition. A suggested default is derived based on the foreign key dependencies stored in the Dictionary or on the key fields of the tables used in the join.

SPFLI			SFLIGHT		
Airline carrier ID	CARRID	00	00	CARRID	Airline carrier ID
Flight connection Id	CONNID	01	01	CONNID	Flight connection Id
Country key	COUNTRYFR			FLDATE	Flight date
From city	CITYFROM			PRICE	Ticket price
Departure airport	AIRPFROM			CURRENCY	Local currency of airline
Country key	COUNTRYTO			PLANETYPE	Plane type
Destination	CITYTO			SEATSMAX	Maximum capacity
Destination airport	AIRPTO			SEATSOCC	Occupied seats
Flight time	FLTIME			PAYMENTSUM	Total of current bookings
Departure time	DEPTIME				
Arrival time	ARRTIME				
Distance	DISTANCE				
Mass unit of distance (k...	DISTID				
Flight type (charter or sc...	FLTYPE				

Definition of a Table-Join without Graphics

All fields contained in the two tables being used are listed with their technical name and long text in the *Link conditions* screen. The two field lists can be scrolled through independently of each other. If two fields are to be linked with the equality relationship, then the same code must be entered for both fields into an entry field assigned to both fields. This code consists of two arbitrary characters (though digits are recommended). This code is similar to the code used for field groups in that its purpose is simply to ensure a unique correspondence between fields.

There are certain restrictions in the database system, which mean that totally arbitrary combinations of fields cannot be linked together. Two fields can be linked only if both fields have the same data type in the Dictionary (including the length attribute). This means that two fields can be linked if they have the same domain. There are separate search functions for each table to assist in the search for suitable fields. Text, domains and data types can be searched for. The text search operates across the technical names of the fields as well as over the long texts. The *Field documentation* function can be used to find information about the technical definition of a field, so as to allow a specific search for suitable fields.

If the same code has been entered for two compatible fields, these two fields are placed in the same line (at the start of the field list) and the fact that they have been successfully linked is indicated by an equals sign. The relationship can be canceled by placing the cursor on one of two fields and calling the *Remove relationship* function, which is to be found on a pushbutton immediately above the equals sign.

When you have finished defining the link conditions for a table pair, you can return to the *Tables in the join* screen by using the *Back* function. When you have finished specifying all the link conditions that you have defined, you can use the *Field groups* function to enter the field group maintenance.

Here you have access to all the functionality that you can use for InfoSets via logical databases, e.g. inclusion of additional tables, definition of additional fields and definition of parameters and selection criteria. Please take note of the remarks made at the end of the [Sequential Datasets \[Seite 278\]](#) section.

On the field group maintenance screen, all tables in the second sub-tree are arranged next to each other at the same level.

It is important to note an important difference to the logical databases: The table join results in a flat table and therefore does not permit an analysis of hierarchical relationships! For this reason additional tables, additional structures and additional fields are always connected to the first table of a join and the only code that exists is for record processing. However, all fields contained in the tables used in the join can be accessed in the WHERE conditions of connected additional tables or the code for additional fields, even though the connection is always made to the first table of the join.

The field group screen contains a *Join* function. This function can be used to call the table join maintenance again, so that changes may be made at any time to the definition of a join or of the link conditions. The only restrictions are that the first table of the join cannot be removed and that other tables can be removed from the join only if none of the fields of these tables are allocated to field groups.

Direct Read

If you want to allow the end-user to generate reports by performing a direct reading on the contents of a table without having to worry about data retrieval, proceed as you would for creating an InfoSet. On the *Title and Database*, specify the table name in the field *Table* and select *Direct read*.

The table must be entered in the ABAP Dictionary and must exist in the database.

Here, you can also choose any of the options available when maintaining InfoSets that use logical databases, for example linking additional tables, as well as defining additional fields, parameters and selection criteria. Please take note of the remarks made at the end of the [Sequential Datasets \[Seite 278\]](#) section.



To read an additional table directly, use an ABAP Open SQL loop of the form

```
SELECT * FROM <table>.  
...  
ENDSELECT.
```

In the following cases, a WHERE condition is automatically appended to the SELECT statement:

1. In InfoSets where selection criteria are defined to refer to the table.
2. In queries where additional selections are defined for fields of the table.

Sequential Datasets

If you want to generate reports from a sequential dataset without having to worry about data retrieval, create an InfoSet by selecting the *Sequential dataset* field on the *Title and Database* screen, enter a file name and a structure created in the ABAP dictionary.

Then, enter the sequential dataset file name you want to read. Its record structure should correspond to the structure specified. You can specify any file name acceptable to the ABAP statement READ DATASET. If you generate a query in this InfoSet and start it online, the file name you specify is displayed as a default value on the selection screen.

Reading from a sequential dataset is performed in binary mode.

You can define field groups on the following screen. Here, you can choose any of the options available when maintaining InfoSets which use logical databases, for example linking additional tables, as well as defining additional fields and structures, parameters and selection criteria, DATA code and START/END code.

When linking additional tables, additional structures and defining additional fields, however, you should be aware of the following special features:

- If you define the InfoSet without a logical database, you cannot make any assignments to a database table.
- The sequence you specify for sections of code involving additional fields, additional structures and additional tables is important.

When defining an InfoSet without a logical database, it obviously makes no sense to specify code for the GET and GET LATE events of a logical database table. Instead, you specify code for record processing. This option is described in the section [Special Features \[Seite 282\]](#).

Retrieving Data with Programs

If you want to make evaluations for datasets, but SAP Query automatic data retrieval is not sufficient for the task, create an InfoSet by selecting the field *Data Retrieval with Program* in the *Title and Database* screen, then specify a program name and the name of a structure. This structure must reflect the record structure of the dataset you want to evaluate.

If you use the component *Maintain Queries* to create a query for this InfoSet, SAP Query uses the report as a model when generating the query report; the model report itself thus remains unchanged.

Before generating the InfoSet for the first time, you must create the model report yourself in the ABAP editor. The model report has to be syntactically correct and have the same definition for fixed point arithmetic as the InfoSet. It is not, however, intended that the model report itself be executed.

The structure of this model report and the sequence of its parts are regular. The main structure of a model report is given below.

Report xxxxxxxx.	
Tables tab.	Definition of Dictionary structure used to set up the InfoSet. This structure must contain the records you want to evaluate.
Parameters.	Definition of parameters, selection criteria, and fields.
Select-Options:...	
DATA:...	
DATA: BEGIN OF itab OCCURS xxx. INCLUDE STRUCTURE tab. DATA: END of itab.	Definition of an internal table <itab> with structure <tab> which provides the records you want to evaluate.
* <Query_head>	This comment must always appear after your data declarations.
* Code to define the table itab, if such a table is used. * Beginning of a loop to retrieve each record and place it in the structure tab (SELECT, DO, LOOP, ...) * Code for formatting data (if necessary)	
* <Query_body>	This comment must always be the last 'statement' in the loop. The data must be available in structure tab.

Retrieving Data with Programs

<pre>* End of data retrieval loop for individual records (ENDSELECT, ENDDO, ENDLOOP;...)</pre>	
--	--



If the model report components are in the wrong sequence, SAP Query may generate meaningless reports. The character strings of the two comment lines *<QUERY_HEAD> and *<QUERY_BODY> are fixed to start immediately after the character "<", but the system does not distinguish between upper and lower case.



Generating the InfoSet is possible only if the model report exists, conforms to the above conventions, is free of syntax errors and has the correct fixed point arithmetic setting.

When you generate query reports which use a model report, the generated reports inherit the attributes of the model report. In the end, this allows you to retrieve data by using a logical database. However, you should only make use of this option in special cases, since if you are using a logical database, it is better to set up the InfoSet this way also.

Below is a model report which uses the SELECT statement:

```
*-----*
*      data retrieval program for functional area FLDP      *
*-----*

report aq00fldp.

tables saplane.

select-options type for saplane-planetype.

*<Query_head>

select * from saplane where planetype in type.

*<Query_body>

endselect.
```

Below is a model report which uses the LOOP statement:


```
*-----*
*      data retrieval program for functional area FLDX      *
*-----*

report aq00fldx.

tables: saplane, indx.

data: planedata like saplane occurs 100 with header line.

select-options type for planedata-planetype.

*<Query_head>
import planedata from database indx(pl) id 'PLANEDAT'.
loop at planedata where planetype in type.
  move-corresponding planedata to saplane.

*<Query_body>
endloop.
```

The data is retrieved and placed in an internal table which is filled by an import from the INDX. In the LOOP structure, each line must be read from PLANEDATA into SAPLANE because the InfoSet was created via the structure SAPLANE and the query expects the data in a field string called SAPLANE.

You can see all the options available for you to use with InfoSets using logical databases in the screen for InfoSet maintenance, for example connecting additional tables and the definition of additional fields, and so on. Make a special note of the special features called [Sequential Datasets \[Seite 278\]](#) at the end of the section.

InfoSets which retrieve data using a program, offer you a wide range of options. In a model report, there are no restrictions on how you should organize the retrieval of your data. Therefore you can use very complex algorithms. SELECT statements that read on a cross-client basis can also be used (option CLIENT SPECIFIED).

Special Features

InfoSets using logical datasets and InfoSets without underlying datasets have largely the same structure. Differences only occur if the situation absolutely demands it. A logical database is a hierarchical structure of tables, whereas an InfoSet without an underlying database is based on a single (sequential) table. You can think of an InfoSet without an underlying database as a special kind of InfoSet which uses a logical database consisting of a single table.

The facilities and options available for creating both types of InfoSet are also the same, apart from the structure of the data to be processed. They include:

- Field group structure
- Options for modifying texts
- DATA Code
- START-OF-SELECTION code
- END-OF-SELECTION code
- Parameter
- Selection criteria

The facilities for InfoSets without an underlying database are simpler as a result of the simpler structure of the data to be processed. This applies to:

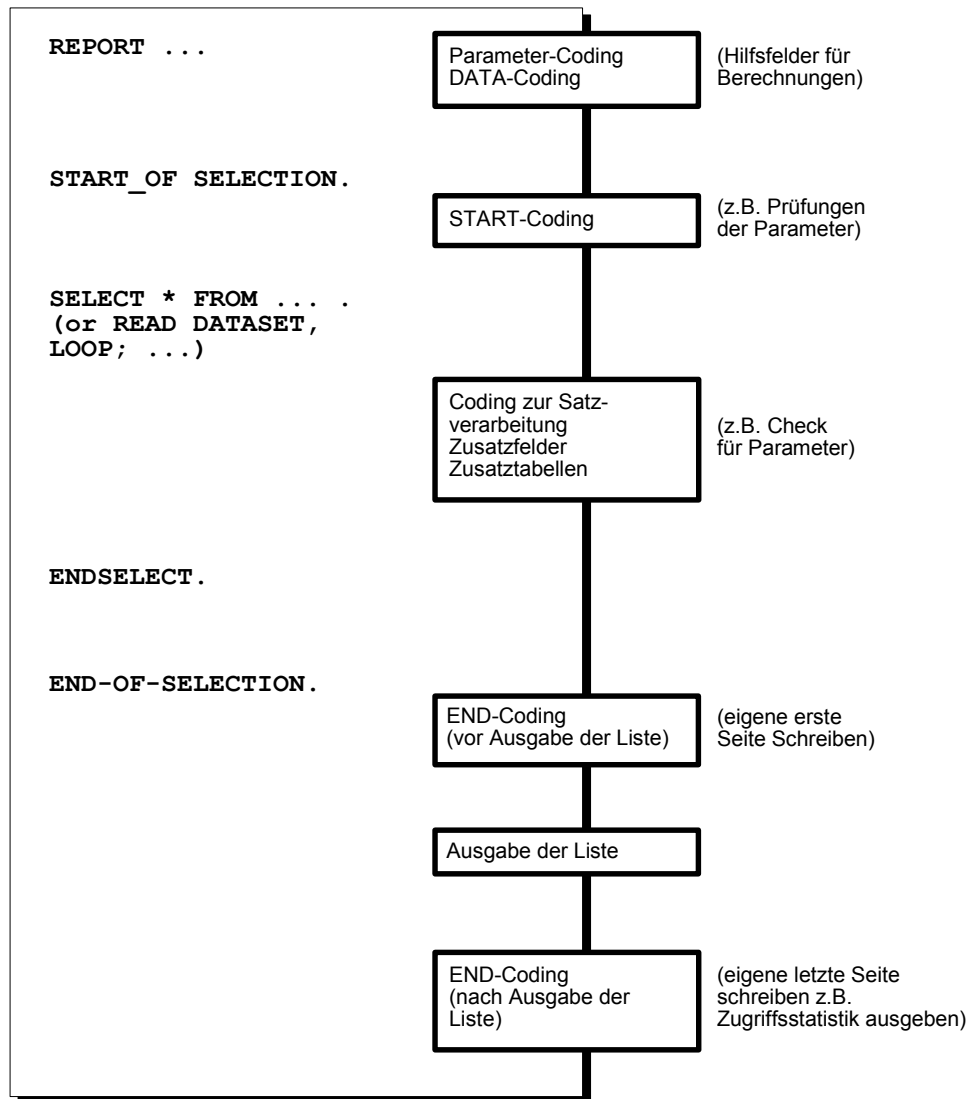
- Additional fields

Additional fields do not need to be assigned to a particular table. However, the sequence you specify is important because it allows you to control the flow. The code lines for an additional field should appear in the section of a generated query report used to process each data record (table entry) read.
- Additional tables

Additional tables are not assigned to a particular table either. The sequence is just as important as it is with additional fields.
- GET and GET LATE code

Since the events GET and GET LATE are closely associated with logical database tables, they do not occur at all in query reports for InfoSets without an underlying database. Instead, there is record processing code. This is inserted in generated query reports at the point (event) where loop processing of data records (table entries) usually takes place. The position it occupies depends on the sequence you specify. There is no equivalent to the GET LATE code, since no data hierarchy exists in this case.

The diagram below shows the sequence of sections of code you can modify in a query report for an InfoSet without an underlying database.



Code for Record Processing

If you want to perform calculations or other operations not associated with additional fields when processing data records in a loop, use the code for record processing.



The code contains statements which check whether the record just read is to be processed according to the parameters or selection criteria.

The code is a standard component of every query report generated for an InfoSet without an underlying database. It is transparent to the user when maintaining queries.

If you are on the field group maintenance screen, choose *Goto* → *Code* → *Record processing*. This takes you to a screen which displays any existing code for record processing. It also displays or proposes (if new) the allocated sequence.



You can also reach record processing code using the *Extras* function (see [GET/GET LATE Code \[Seite 262\]](#)).

The code for record processing is placed, together with the code for additional fields and SELECT statements for additional tables, in the section of the generated query report containing the loop processing of data records. The sequence specifications control the order within this section. Usually, the code for record processing is used for initialization purposes and appears before the additional fields and SAP tables. For this reason, the system proposes 0 as the sequence number.

On the screen for defining the code for record processing, you can thus perform the same operations as when you define the GET/GET LATE code (see [GET/GET LATE Code \[Seite 262\]](#)).



To ensure that query reports can be generated without problems, it is important that it is clear which fields are accessed in every piece of code. If some code contains ABAP statements that access certain fields implicitly, then an ABAP FIELDS statement must also be used in the code to ensure that all fields used are explicitly named. You can find an example of this and further information in [Access Optimization in Queries \[Seite 342\]](#)

Functions for Managing User Groups

Within a team or department, the problems you solve using SAP Query reports are fairly similar and involve the same InfoSets. For this reason, SAP Query has been designed to allow you to combine users with similar concerns together in user groups.



The information is intended for system administrators.

The characteristics of a user group include:

- Grouping together users with similar tasks to perform
Each member of a user group can execute any queries within that group. However, only those user group members possessing the required authorization can maintain the queries (see [SAP Query Authorizations \[Seite 81\]](#)).
- Assigning InfoSets
All InfoSets covering one task area are assigned to one user group. The members of this user group can define only those queries in the InfoSets assigned to their group.

As a rule, a user is part of one user group. But it is also possible that users with several work areas are also members of several user groups.

To manage user groups, you use the component *Maintain User Groups*.

Displaying User Group Directories

User Group Directory

For a list of existing user groups, select *Environment* → *Directories* → *User groups* or display a list of possible entries for the relevant input field. The resulting output contains the user group ID and description for each user group.

If you choose a user group from this list, the system returns you to the initial screen and displays the user group you have selected in the appropriate field.

Directory of Users per User Group

For a list of users and the user groups to which they are assigned, select *Environment* → *Directories* → *Users/user groups*. The selection screen of this report allows you to restrict the list to individual users or user groups. If you do not specify any restrictions, the system generates a list of all users belonging to any user groups.

You can scroll through the list and choose a user group with the *Choose* function.

InfoSet Directory

For a list of existing InfoSets, select *Environment* → *Directories* → *InfoSets* or display a list of possible entries for the relevant input field. This is the same list you see when you call the corresponding function in the component *Maintain InfoSets*. If you use the choose an InfoSet from the list, the system returns you to the initial screen and displays the InfoSet in the field *InfoSet*. If you want to know which InfoSets are assigned to a particular user group, display the user group. See [Managing User Groups \[Seite 287\]](#) for more information.

Managing User Groups

This section describes the functions available in SAP Query for managing user groups.

Displaying User Groups

Displaying User Groups

Whenever you need information about an existing user group, enter the name of the user group on the initial user group maintenance screen and choose *Display*. A long text describing the task area for which the user group is intended. The author and the person who made the last set of changes to the user group are also displayed.

To see which users belong to a user group, choose *Description*. The resulting list also shows you which InfoSets are assigned to the user group.

Creating User Groups

To create a user group, proceed as follows:

1. Enter a name in the field *User group*.

A user group name can contain up to 12 characters. This must be a user group name which does not already exist.

2. Choose *Create*.

You then see a screen where you enter a long text for the user group. Since the long text of a user group is seen by all end-users, it should reflect the nature of that user group more precisely than the group's name.

The screenshot shows a SAP dialog box titled "User Group FLIGHT: Create or Change". It contains the following fields and values:

User group	FLIGHT	Flight data evaluations
Author	ABAP	02.09.1999
Last changed by		

At the bottom of the dialog box, there are two buttons: a green checkmark icon followed by the text "Save", and a red X icon.

3. Save your entries.

Assigning Users and InfoSets

Assigning Users and InfoSets

1. Enter a name in the field *User group*.
2. Choose *Assign users and InfoSets*.
3. Enter the names of those users that should belong to this user group.

User group	FLIGHT	Flight data evaluations
Users and change authorizations for queries		
BARZEWSKI <input checked="" type="checkbox"/>	CUSHMAN <input type="checkbox"/>	EDINGERM <input checked="" type="checkbox"/>
MAIER <input type="checkbox"/>	SCHWINN <input checked="" type="checkbox"/>	KESSLERK <input checked="" type="checkbox"/>

If you do not want to enter the names of users individually, choose *Settings → With selection*. You then see a list of all users with a user master record in your system. In this list, select the desired names.

If you want to enter the names yourself, select *Settings → Without selection*.

In order to change queries, a user needs an authorization for authorization object S_QUERY with the value Change (see [SAP Query Authorizations \[Seite 81\]](#)). These change authorizations can be revoked for each individual user of a user group.

By de-selecting the checkbox behind a user's name you can revoke the change authorization of individual users. In order to revoke change authorization for all users, choose *Delete all change authorizations*.



Change authorization can, however, only be revoked, if the user in question actually has a corresponding authorization in the user master record. Checkboxes behind user names that have no change authorization according to S_QUERY are grayed out and cannot be selected.

4. Assign one or more InfoSets to the user group.



Assigning InfoSets to user groups is a requirement for defining queries. You can define queries only for those InfoSets assigned to user groups in which you are active.

Each InfoSet can be assigned to several user groups.

To display a list of all InfoSets, select *Assign InfoSets*.

You assign InfoSets to a user group, by selecting the ones you require.

5. Save your entries.

Assigning Users to User Groups

To assign a user to one or more user groups, proceed as follows:

1. Enter the user name in the field *User* on the initial screen.
2. Choose *Assign user*.
3. In the list of user groups, select the groups to which you want to assign the user.
4. Save the assignment.



In user group lists you can revoke change authorization for queries from those user groups to which a particular user is assigned. For further information, see [Assigning Users and InfoSets \[Seite 290\]](#).

Assigning InfoSets

Assigning InfoSets

You can only define queries for those InfoSets assigned to your current user group. If you want to assign new created InfoSets to several user groups, without having to edit each user group manually, you can use the *Assign InfoSet* function on the initial screen of the component *Maintain User Groups*.



There is a similar function in the component *Maintain InfoSets*.

To assign an InfoSet to several user groups, proceed as follows:

1. Enter the InfoSet name in the field *InfoSet* on the initial screen.
2. Choose *Assign InfoSet*.
3. In the list of user groups, mark the groups to which you want to assign the InfoSet.



You can also use the *Assign InfoSet* function if the InfoSet has already been assigned to some user groups, but you now want to change the assignment. The user groups to which the InfoSet has already been assigned are flagged.

Copying User Groups

To copy a user group, proceed as follows:

1. Enter the name of the user group in the field *User group* on the initial screen.
2. Choose *Copy*.
3. Enter the name of the new user group.
4. By selecting the relevant field, specify whether you want to copy the queries of the user group to be copied.

The copied user group contains the same users as the original. It is assigned to the same InfoSet. If you opted to copy the queries, it also contains all the queries from the original user group. Otherwise, it contains no queries.

Renaming User Groups

Renaming User Groups

To rename a user group, proceed as follows:

1. Enter the name of the user group in the field *User group* on the initial screen.
2. Choose *Rename*.
3. Enter the new name of the user group.

The following objects have to be locked before this function can be executed:

- User group catalog
- Query catalog of the user group being renamed
- Queries of this user group
- Query list catalog of the queries of the user group

The function is executed only if all these were able to be set. As long as these locks are set, other users may be prevented from continuing their work. Note that renaming a user group can be a very time-consuming activity!

Deleting User Groups

To delete a user group, proceed as follows:

1. Enter the name of the user group in the field *User group* on the initial screen.
2. Choose *Delete*.

First, the system asks you to confirm your intention to delete. It then checks whether any queries exist for the user group. If so, it gives you an opportunity to delete them.

Deletion is possible only if no further queries exist for the user group.

Changing User Groups

Changing User Groups

To change a user group, proceed as follows:

1. Enter the name of the user group on the initial screen.
2. Choose *Change*.

A dialog box appears where the user group's long text can be altered.

You can now change which user groups your users and InfoSets are assigned to by using the functions *Assign users and InfoSets*, *Assign Users* or *Assign InfoSet*.

Query Areas

Query areas were created to allow you to fulfill varying requirements with SAP Query.

A query area contains a set of query objects (queries, InfoSets, and user groups) that are discrete and consistent.

You can differentiate between two different query areas, the standard area and the global area. Both query areas provide you with a full range of SAP Query functions.

- **Standard Area**

In the standard query area, all query objects (queries, InfoSets, user groups) are created and managed specifically for each client. Query objects are not attached to the Workbench Organizer, this means that they cannot be created and transported according to standard correction and transport procedures. This is a big advantage for end users that want to develop queries (ad-hoc reports) in their own client that are not meant for use in the rest of the system. Query objects can still be transported, however the transport process requires manual preparation and is not automatically initiated (export and import takes place using Query transport tools). For further information, see [Transporting Standard Area Objects \[Seite 306\]](#).

- **Global Area**

Query objects in the global area are cross-client, that is they are available throughout the whole system and in all clients. Query objects in the global area are connected to the Workbench Organizer. They can be created and transported using the normal correction and transport procedures.

Transport is initiated automatically and no manual preparation is required. The global query area is therefore well suited for centrally developing queries meant for use and distribution throughout the system. All query objects delivered by SAP (from Release 4.0) are located in the global query area.

Both the global and the standard query areas contain discrete and consistent numbers of query objects. No relationships of any sort can exist between objects from different query areas. For example, you cannot create a query in the standard area using an InfoSet from the global area.



Each query area can be viewed as a discrete namespace for query objects. This means that objects can exist in different query areas that have the same name but different meanings.

Global Area Naming Conventions

How to name queries, user groups, and InfoSets has already been discussed in the previous sections. When naming global area query objects, certain prefixes can be used.



You may only use a particular prefix after first having purchased its corresponding license.

A prefix is composed of '/prefix/' and precedes the actual object name. The / symbols before and after the prefix name are actually part of the prefix. A prefix can contain up to 10 characters including the two / symbols.

Pay attention to the following guidelines:

- InfoSet names can take the form '/prefix/InfoSet'.

The entire name of the InfoSet including prefix can contain up to 24 characters.

Query Areas

- User group names can take the form '/prefix/user_group'.
The entire name of the user group including prefix can contain up to 12 characters.
- Query names take the form 'query'.
Query names have no prefix of their own, instead they take the prefix of their user group.
A query name can contain up to 14 characters.
- Prefixes may only be used with InfoSets and user groups from the global area. The Workbench Organizer checks to see if prefixes are used correctly.

The individual query object maintenance transactions check the name syntax described above. You can only work using name prefixes with query objects in the global query area. This is significant because it is in this area that query objects from SAP are inserted at PUT if necessary. All query objects delivered by SAP have the prefix '/SAPQUERY/', which has been reserved for them.



When using prefixes in the global area, ensure that objects whose names begin with a prefix belong to the development class with the same prefix. The Workbench Organizer checks to see if this condition has been fulfilled.

When creating queries for a user group whose prefix belongs to SAP, a business partner or another R/3 customer, you must pay special attention to the fact that queries 'inherit' their prefixes from their user groups. These kinds of user groups can in turn be transported into your system using a PUT or other transport method. These queries then belong to the objects found in the namespace determined by the user group's prefix.

If, for example, a query is created in the user group '/SAPQUERY/xx', then this query inherits the prefix '/SAPQUERY/'. It seems, therefore, that the query belongs to those objects delivered by SAP. This in turn could lead to the query being overwritten during the next transport. Therefore, it is recommended that no new queries be created in these user groups and that you use your own user groups when creating new queries.

Changing Query Areas

You can change query areas from each query object maintenance component by choosing *Environment* → *Query areas*. A window appears containing both query areas and their long texts.

Use *Choose* to choose the query area that you want. If you choose to work in the global area, this is displayed on the initial screen of the maintenance component. In the standard area no such text is displayed. The same maintenance component functions are available in both query areas.

You can display technical information about a query area by choosing *Information* on the screen where you chose which query area you wanted to work in. A dialog box appears where a long text is displayed along with information about whether or not the objects are client specific or linked to the Workbench Organizer.

Assigning Development Classes in the Global Area

Due to the fact that query areas are linked to the Workbench Organizer, a development class must be designated when query objects are created. Query objects are entered into a correction request whenever they are created or changed.

In the global area, you can classify query objects as local objects (using a temporary development class, usually \$TMP). (This is the same as creating a query object in the standard area). There are several conditions that you should pay attention to when designating or changing development classes:

- All query objects in the global area have to be assigned to a development class (temporary development classes included). If the development class is not temporary, that is to say

transportable, then the object must be entered in a correction request when it is being created or changed.

- User groups and InfoSets can be assigned to any development class you want.
- Queries can only be assigned to non-temporary development classes if their corresponding user groups and InfoSets are also assigned to non-temporary development classes. If, when you are creating a query, it is determined that either the InfoSet or user group assigned to that query is part of a temporary development class, then your query will automatically be assigned to development class \$TMP. Whenever this happens, no dialog box asking you to determine a development class will appear.



In order to simplify matters, it is recommended that you always assign user groups and InfoSets in the global area to transportable development classes. In this way, all queries can be assigned to any development class you want.

All query objects in the global area that are assigned to non-temporary development classes must be entered in a correction request when being created or changed. An exception occurs with customizing settings. Changes like assigning users to user groups and InfoSets to user groups can be made without having to be entered in a correction request. Transports made using the Workbench Organizer do include InfoSets' user group assignments, not however users' user group assignments.

You can change development classes in the various query object maintenance components by choosing either *Query* → *More functions* → *Change development class*, *InfoSet* → *More functions* → *Change development class* or *User group* → *Change development class*. However, in accordance with the rules for assigning development classes formulated above, the following restrictions apply:

- Changing a user group from a non-temporary development class to a temporary development class only makes sense if all of the user group's queries are assigned to a temporary development class.
- Changing an InfoSet from a non-temporary development class to a temporary development class only makes sense if all of the InfoSet's queries are assigned to a temporary development class.
- Changing queries from a temporary development class to a transportable development class only makes sense if the InfoSet areas and user groups they are assigned to are in a transportable development class themselves.

The function *Change development class...* checks to see if you are allowed to change an object's development class and displays a warning if this is not possible. You may then change the development class.



For technical reasons it is possible to make changes that conflict with the restrictions listed above. Therefore the system checks automatically to see if all changes made are acceptable. If a conflicting change has been made, a warning is displayed asking you to reconsider the change. If this were not the case, you would run the risk of having inconsistent datasets in all receiving systems after transport.

You can no longer change a development class when an object belongs to a transportable development class and has already been transported. This means that all development class changes described above must be made prior to transporting the object.

A development class change may also be necessary after a query object has been renamed. You must also ensure that the development classes of user groups and InfoSets that have been

Query Areas

renamed still lie within the accepted name spaces for the new names. In this case it is advisable to choose an appropriate development class when you are renaming the object. With user groups the individual queries within the user group must also be assigned new development classes.

Renaming InfoSets and User Groups in the Global Area

When using the function *Rename* with InfoSets and user groups be aware of the fact that in addition to the InfoSet or user group, all dependent queries must also be renamed. If several InfoSets or user groups have been renamed, a series of queries must be included in a correction request in addition to the InfoSets and user groups. The renaming process is only actually finished when all objects necessary have been entered into a correction request.

Copying Query Objects between Different Query Areas

In order to copy query objects from one query area to another a special procedure must be followed to ensure that the datasets of each query area remain intact and consistent. During transfer you must check to see that the object being transferred can be inserted in the new dataset without upsetting the consistency of the latter. If you think of the global area as a single client, then the whole copying process, from the standard area to the global area and vice versa, corresponds to object transport from one client to the next. Thus, you copy query objects from one query area to the next in much the same manner as you copy objects within the standard area.



Only those users in possession of the necessary transport authorization (InfoSet and user group maintenance authorization) can copy query objects from one query area to another.

Global Area Query Variants

If you want to transport query variants within the global area, these variants must be created as system variants. System variant names begin with either SAP& or CUS&. Other kinds of variants cannot be transported. (Please see also the variant documentation available on the initial variant maintenance screen).

Maintaining the Additional Function Pool

All interactive functions for adjusting query lists (*Display as table*, *Download to file*, *ABC analysis*, and do on) work according to the same principles: Data from a sublist together with a description of this data is passed to another program (function module) using an interface. Enhancement SQUE0001, which is intended for customer use, also works according to this principle (see [Enhancement SQUE0001: Private File \[Seite 334\]](#)).

The number of functions that can be attached to query lists in this way is not limited. The *Additional function pool* acts as a container for these functions.

The *Additional function pool* function bundles an arbitrary number of interactive functions for single-line sublists that work according to the principles for passing data outlined above. Each of these functions must be implemented using a function module with a defined interface. Which and how many of the functions you want to make available to the user can be determined by using the maintenance component for the *Additional function pool*. You call the additional function pool maintenance screen (transaction SQ09) from the initial screens of both the InfoSet maintenance component and the user group maintenance component by using either the function *Environment* → *Additional function pool* or its corresponding pushbutton.

When you call the additional function pool maintenance component, a screen appears listing all of the functions already contained in the additional function pool. For further information, refer to the section on the [Additional Function Pool \[Seite 130\]](#).

You can also enter as many new functions or functions of your own to the *Additional function pool* as you like.

Each function that you want to call using the additional function pool must be:

1. Implemented as a function module with a fixed interface
2. Entered in specific tables in the R/3 System
3. Released for use in the additional function pool (activated)



The interface required for function modules used in the additional function pool is the same interface as used in function module RSAQ_XINT_DISPLAY_BASIC_LIST. This function module is one of the functions for the additional function pool delivered by SAP.

Whenever you want to use a function module with the additional function pool, you should proceed as follows:

1. Enter the function module as an additional function pool function in InfoSet maintenance using the *Additional function pool* pushbutton (*Create* pushbutton).
2. Activate the function for the additional function pool.



All *Additional function pool* functions delivered by SAP can also be found in the list of additional functions, but are not yet active. You must first activate these functions in order to be able to use them.

Each function contained in the additional function pool is characterized by the following entries:

- Sequence number
- Name
- Activation checkbox
- Long text

Maintaining the Additional Function Pool

- Name of function module

The sequence number you select for the activated functions in this list determines the order in which they are listed in the dialog box (see above).

Each function is identified by a (language independent) name and a (language dependent) long text. Both the name and the long text (description) appear in the aforementioned dialog box. The function module found in the list is called each time you call its corresponding function.

No changes can initially be made when you call the additional function pool maintenance component. In order to make changes, you must first call the *Additional function pool* → *Display* ↔ *Change* function.

In change mode you can activate and deactivate those functions available on the overview screen and determine the sequence of those functions that are active. You activate a function by selecting the checkbox in the *Active* column and can deactivate it by deselecting the same checkbox. Once a function is active, the *No.* column is ready for input. You can enter sequence numbers (whole numbers) here.



Activating/deactivating functions and determining their sequence on this screen is actually a technical process similar to Customizing settings. No change request is made for these settings in the Change and Transport Organizer. This also means that these settings have to be remade in every SAP system.

In order to enter a new function in the additional function pool, choose *Create* on the overview screen. A dialog box appears where you enter the function name, long text, and the name of its function module. You may only enter the names of those function modules that have the right kind of interface. Function modules may only be entered once in the additional function pool.

Once a function has been entered in the pool, only its long text may be altered. To do this, select the function whose long text you want to change and choose *Edit* → *Change text*. The same dialog box is displayed that was displayed when you added the function to the function pool, however, now only the long text field is ready for input.

In order to delete a function from the additional function pool, select the function you want to delete and choose *Edit* → *Delete*. A dialog box appears asking you to confirm that you really do want to delete the entry from the additional function pool, choose yes to delete the entry. The function module, however, remains intact.

Each entry in the additional function pool constitutes a unique transport object (AQXI). The creation, change and deletion of this kind of entry are monitored by the Change and Transport Organizer. Thus, whenever you assign these entries to non-temporary development classes, they are distributed to other SAP systems according to normal transport procedure.



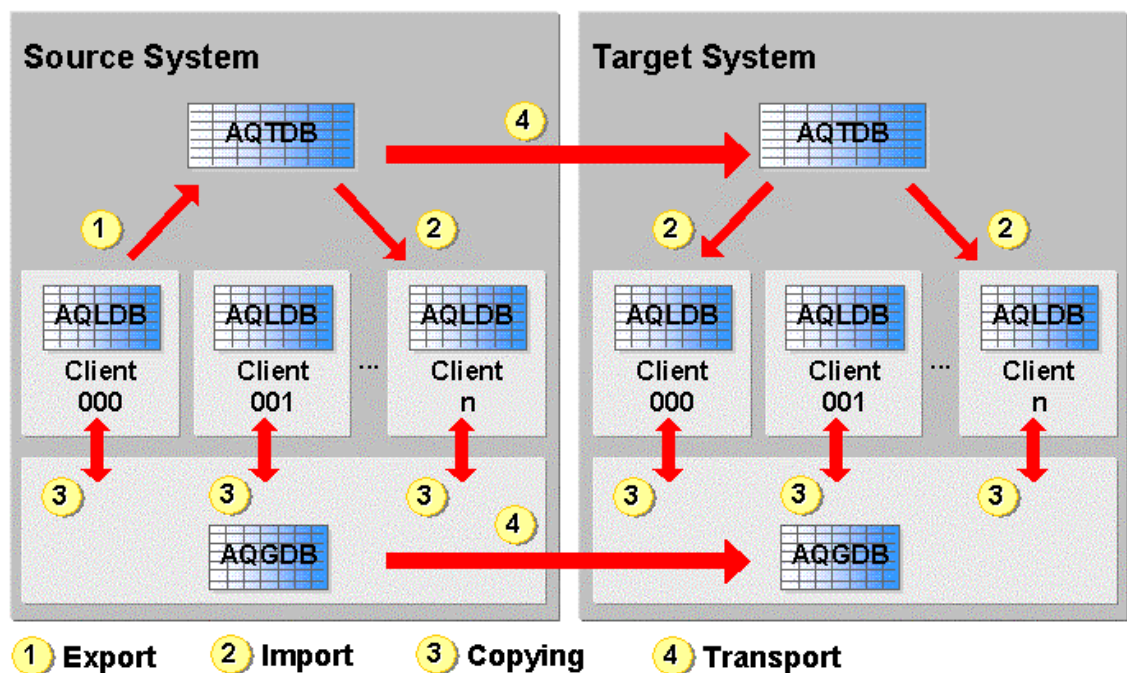
Because all additional function pool settings are valid in both query areas, all changes made to the additional function pool take effect in both the global area, and in all clients in the standard area of your SAP system.

Transporting Query Objects

Query objects are transported in different ways according to the query area in which they were created.

In order to know which transport options are available, you must first understand how query objects are created.

- **Standard Area**
Query objects are stored in the client-specific table AQLDB. They are not connected to the Change and Transport Organizer.
- **Global Area**
Query objects are stored in the cross-client table AQGDB. They are connected to the Change and Transport Organizer.



Global area objects can be transported into other systems. Standard area query objects can not only be transported to other clients within their own system, but into all clients of other systems as well. In addition, query objects can be transported from the global query area to the standard query area and back within the same system.



Transports are normally performed by the system administrator, not by end-users. For this reason, you need the appropriate authorizations (see [SAP Query Authorizations \[Seite 81\]](#)).

For further information, refer to the following sections.

Transporting Global Area Objects

The global area contains cross-client objects. Therefore, transporting them always means moving them to another system.

In the global area, an object's development class determines whether or not it can be transported. Local objects (those in temporary development classes) cannot be transported. All other objects are entered into a correction request whenever they are created or changed. Transport then takes place in the normal manner using the Change and Transport Organizer (transport request release).

Whenever an object is transported using the Change and Transport Organizer, you are unable to check the Query objects being transported for compatibility with the target system. Thus it is possible for inconsistencies to crop up in the target system's Query datasets. A typical inconsistency of this sort would be transporting a Query, whose corresponding InfoSet does not exist in the target system. Therefore, when transporting Query objects in the global area, it is important to check and see if all current dependent objects already exist in the target system. If this is not the case then these objects must be transported as well.



Use the menu option *Goto → More functions → Consistency Check* in the InfoSet maintenance component to check Query datasets in the target system for consistency after transport. You can choose which objects to check and a log is generated containing a list of any inconsistencies discovered.

If inconsistencies are discovered, the action you take to eliminate them depends on the type of inconsistency. Some inconsistencies must be eliminated in the system, for example if users appear in a user group that no longer has a user master record. Other inconsistencies can only be eliminated by completely re-transporting the objects from the source system. A good example of this kind of inconsistency is the Query transport without its corresponding InfoSet, mentioned at the beginning of this section.

If no catalog entry is found during a consistency check or there are objects in the catalogs that no longer exist, you can use the *Goto → More functions → Reconstruct catalogs* function to reconstruct these catalogs. A selection screen is displayed where you can choose which catalogs you want to reconstruct.

Query variants can only be transported if these variants have been created as system variants (see *Global Area Query Variants* in [Query Areas \[Seite 297\]](#)). All other variants cannot be transported.

The following entries are included in a correction request when any global area Query objects with transportable development classes are altered:

1. User groups

R3TR AQBQ bbbbbbbbbbbb

This transport object only contains the catalog entry for a user group. The user assignments and InfoSet assignments of this user group are neither recorded nor transported.

Changes to the user/user group assignment or the InfoSet/user group assignment can be made without having to be entered in a correction request. When the transport request is released, it includes InfoSet/user group assignments, not however users/user group assignments.

2. InfoSets

R3TR AQSG sssssssssssssssssssssss

This transport object contains the catalog entry for an InfoSet, its definition, and its user group assignments. When the transport request is released, the InfoSet's current user group assignments are transported as well.

3. Queries

R3TR AQQU bbbbbbbbbbbqqqqqqqqqqqqqq

This transport object contains the catalog entry for a Query, its definition, and the Query itself.

4. System Variants of Queries

[illegible]

This transport object contains exactly one system variant of a single Query. System variants can only be created when their Query is not assigned to a temporary development class. You should also pay attention to the fact that this transport object cannot be locked and can thus show up in different correction requests simultaneously. If the Query is transported together with the system variants because of this, make sure that the system variants are included in the same correction requests as the Queries.

Changes to the user/user group assignment or the InfoSet/user group assignment can be made without having to be entered in a correction request. When the transport request is released, it includes InfoSet/user group assignments, not however users/user group assignments.

Transporting Standard Area Objects

This chapter describes how to transport Queries, InfoSets and user groups found in the standard area. Standard area queries are client specific. Transporting these types of objects can mean that these objects are being transferred to another client within the same system or that they are being transported into a totally different system.

To access the standard area transport tool, call the component *Maintain InfoSets* or *Maintain User Groups* and choose *Transports*.

For further information, refer to the following sections.

General Transport Description

Queries, InfoSets and standard area user groups are usually stored in the client-specific table AQLDB. When transporting these objects, however, you use the transport table AQTDB, which is not client-specific.

You can transport objects either

- from one client (the source client) to another client (the target client) within the same SAP System
or
- from one client (the source client) in one R/3 System (the source system) to another client (the target client) in another R/3 System (the target system).

In both cases, you use similar procedures. For a graphical overview, see [Transporting Query Objects \[Seite 303\]](#).

There are two options for performing transports. The first of these involves using the transport table AQTDB and the transport actions *Export* and *Import*.

- You must first export the objects for transport from the table AQLDB to the transport table AQTDB.

This creates a transport request for the transport system that includes all entries created by the export from table AQTDB (transport dataset). The request covers all the entries from table AQTDB (transport dataset) generated by the export and has a name which corresponds to transport system conventions (sysKnnnnnn).

The name of the transport request is given in the export log.

In the table AQTDB, the transport dataset is stored under a key which matches the name of the transport request.

- If you are transporting within one R/3 System (that is only from one client to another), you can also import the transport dataset from the target client.

In this case, you use the name of the transport request which is specified during the export and matches the key of the transport dataset in the table AQTDB.

During this import, the system reads the transport dataset from the transport table AQTDB into the table AQLDB. At the same time, it performs numerous checks which ensure that no inconsistencies occur in the table AQLDB.



Both export and import are functions of the SAP Query transport tool.

- If you are transporting from one R/3 System to another, you must use the transport system to release and export the transport request created by the export.

This procedure transfers the transport dataset from the transport table AQTDB of the source system to the transport table AQTDB of the target system. Then, you can perform an import from the target client of the target system, as described above.



The transport datasets created in the transport table AQTDB by exports are not deleted after successful imports unless you specify this explicitly beforehand. For this reason, you cannot import a transport dataset several times, for example to transport an InfoSet to several clients.

The second transport option involves transferring files via the R/3 download/upload interface. To do this, you use the transport functions *Download* and *Upload*.

General Transport Description

- You must first download the source client objects for transport from the table AQLDB to a file. When doing this, you specify the file name and exactly where it is stored in the file system. A file can only contain one transport dataset, meaning that any existing file is overwritten. Downloading a transport dataset to a file effectively means that it cannot be read back into the table AQLDB.
- Then, you can upload the transport dataset stored in the file. Here, the transport dataset is read and inserted in the table AQLDB. As with an import, the system performs numerous checks which ensure that no inconsistencies occur in the table AQLDB.

After this general overview, you can familiarize yourself more closely with the individual options offered by the transport tool. On the initial screen of the component *Maintain InfoSets* or *Maintain User Groups*, select *Transports*. You then see the following screen:

Transport action	<input checked="" type="radio"/> Export	<input type="radio"/> Download	<input type="radio"/> Display
	<input type="radio"/> Import	<input type="radio"/> Upload	<input type="radio"/> Delete
	<input type="radio"/> Copy Standard Area -> Global Area		
	<input type="radio"/> Copy Global Area -> Standard Area		
<input checked="" type="checkbox"/> Test run	<input type="checkbox"/> Overwriting allowed (only with import/upload/copy)		
<input type="checkbox"/> Transport query variants (only with export/import/copy)			
<input type="checkbox"/> Transport the report-reprot interface for queries (only with export/import)			
<hr/>			
<input checked="" type="radio"/> Transport user groups	Import option	REPLACE	
User groups	<input type="text"/>	to	<input type="text"/>
<hr/>			
<input type="radio"/> Transport InfoSets	Import option	REPLACE	
InfoSets	<input type="text"/>	to	<input type="text"/>
<hr/>			
<input type="radio"/> Transport InfoSets and queries	Import option	REPLACE	
InfoSets	<input type="text"/>	to	<input type="text"/>
Queries	<input type="text"/>	to	<input type="text"/>
	Import option queries	REPLACE	
<hr/>			
<input type="radio"/> Transport queries	Import option	REPLACE	
User groups	<input type="text"/>	to	<input type="text"/>
Queries	<input type="text"/>	to	<input type="text"/>
<hr/>			
Dataset with imports	<input type="text"/>	to	<input type="text"/>
<input type="checkbox"/> Delete after successful import			

You must make a number of entries here. First, you choose one of the following transport actions by selecting the appropriate radio button:

General Transport Description

Transport action:	Description:
Export	Generates a transport dataset in the table AQTDB
Import	Reads in a transport dataset from the table AQTDB
Download	Generates a transport dataset in a file
Upload	Reads in a transport dataset from a file
Display	Displays the transport dataset in the table AQTDB
Delete	Deletes transport datasets from the table AQTDB
Copy Standard Area → Global Area	Copies query objects from the standard area to the global area
Copy Global Area → Standard Area	Copies query objects from the global area to the standard area

When you have specified this field and, if necessary, other fields, you can start the transport action by selecting *Execute*. The system always generates a log containing details of the individual actions executed.

Generating Transport Datasets

The following sections contain information about generating transport datasets:

[Transport Types \[Seite 311\]](#)

[Transporting a User Group \[Seite 313\]](#)

[Transporting an InfoSet \[Seite 314\]](#)

[Transporting a Query \[Seite 315\]](#)

To generate a transport dataset, choose one of the transport actions *Export* or *Download* on the initial screen of the transport tool.

If you want to run a test export first, select the checkbox *Test run*. The system then determines which queries, InfoSets and user groups would be transported for the selections entered below on the screen and generates a log. It does not generate a transport dataset in the table AQTDB or a file. You should always make use of this checking facility.

The check box labeled *Overwriting permitted* allows you to make sure that existing objects do not get accidentally overwritten when importing or uploading. If you are doing a test run for an import, then a note will be made in the log if an existing object would get overwritten at any point. If you are not doing a test run and the *Overwriting permitted* check box is not set, then the import of an object will be terminated if the object already exists in the target system and would get overwritten by the import. An existing object will be overwritten only if the check box has been set. A note will be made in the log that an object has been overwritten in this case too.

If you want the transport to include query variants, i.e. variants defined for the query reports, select the checkbox *Transport query variants*. The system then transports all the variants of all queries to the transport dataset. Please note that, to transport variants, you must always use the transport actions *Export* and *Import*. The actions *Download* and *Upload* cannot transport variants. The logs contain details of the variants transported.

Transport Types

There are four different ways you can select the objects (user groups, InfoSets, queries) to be transported. They are covered by the following transport types:

- Transport user groups
- Transporting InfoSets
- Transporting InfoSets and queries
- Transport queries

You first select no more than one of these transport types. Then, you make further specifications according to the transport type chosen. Each transport type has its own input fields.

For each transport dataset, you define an import option according to the relevant transport type. The import option determines the method used to insert the transport dataset in the table AQLDB of the target client during a later import.

There are four import options, each of which has a short form:

Import option:	Abbreviation:
REPLACE	R
MERGE	M
GROUP=ug (ug = User Group Name)	G=ug
UNASSIGN	U

With the transport type *Transport user groups*, all user groups are transported according to the select option *User groups* and the chosen *Import option*. Valid import options are REPLACE and MERGE.

With the transport type *Transport InfoSets*, all InfoSets are transported according to the select option *InfoSets* and the chosen *Import option*. Valid import options are REPLACE, MERGE, GROUP=ug and UNASSIGN.

With the transport type *Transport InfoSets*, all InfoSets are transported according to the select option *InfoSets* and the chosen *Import option*. Valid import options are REPLACE and MERGE. Furthermore, all queries are transported according to the select option *Queries*, which use the InfoSet, for each of the InfoSets selected in this way. The query user groups are of no relevance here. The only valid import option for these queries is REPLACE. This transport type facilitates the transport of a modified InfoSet and all associated queries.

With the transport type *Transport queries*, all queries are transported according to the select option *User groups* and the select option *Queries* as well as the chosen *Import option*. Valid import options are REPLACE and GROUP=ug.

If, for example, you want to export the InfoSet FLBU and all the associated queries, you must make the following entries:

Transport Types

Transport action	<input checked="" type="radio"/> Export	<input type="radio"/> Download	<input type="radio"/> Display
	<input type="radio"/> Import	<input type="radio"/> Upload	<input type="radio"/> Delete
	<input type="radio"/> Copy Standard Area -> Global Area		
	<input type="radio"/> Copy Global Area -> Standard Area		
<input checked="" type="checkbox"/> Test run	<input type="checkbox"/> Overwriting allowed (only with import/upload/copy)		
<input type="checkbox"/> Transport query variants (only with export/import/copy)			
<input type="checkbox"/> Transport the report-reprot interface for queries (only with export/import)			
<hr/>			
<input type="radio"/> Transport user groups	Import option: REPLACE		
User groups		to	
<hr/>			
<input type="radio"/> Transport InfoSets	Import option: REPLACE		
InfoSets		to	
<hr/>			
<input checked="" type="radio"/> Transport InfoSets and queries	Import option: REPLACE		
InfoSets		to	
Queries		to	
Import option queries		REPLACE	
<hr/>			
<input type="radio"/> Transport queries	Import option: REPLACE		
User groups	FLBU	to	
Queries	*	to	
<hr/>			
Dataset with imports		to	
<input type="checkbox"/> Delete after successful import			

You get an export log in the export result. The export log shows which objects (queries, InfoSets, user groups) are exported and which import option is used to import each object during a later import. If you export data to the table AQTDB, the export log still contains the name of the generated transport dataset which matches the name of the generated transport request.

Transporting a User Group

When transporting a user group, the system transports all members of that group from the source client. It does **not** transport the assignment of InfoSets to this user group in the source client.

During an import, the import option REPLACE first deletes all members of that group in the target client and then enters the members of that group from the source client.

During an import, the import option MERGE merges all members of that group from the source client with all members of that group in the target client.

You can enter a new user group member in the target client only if a user master record exists.

The assignment of InfoSets to this user group remains the same in the target client.

Transporting InfoSets

When transporting an InfoSet, the system transports the InfoSet and the assignment of that InfoSet to user groups in the source client.

To import an InfoSet, the appropriate logical database must exist in the target client. Moreover, a thorough check is run on the logical database. This check highlights the there are inconsistencies between the InfoSet and the logical database.

During an import, the import option REPLACE first deletes the assignment of that InfoSet to user groups in the target client. It then transfers the InfoSet to the target client and copies the assignment of the InfoSet to user groups from the source client, if corresponding user groups exist in the target client.

During an import, the import option GROUP=bg first deletes the assignment of that InfoSet to user groups in the target client. It then transfers the InfoSet to the target client and assigns it to the user group ug of the target client, if such a user group exists.

During an InfoSet import, the import option MERGE makes sure the assignment of that InfoSet to user groups in the target client is maintained. It then transfers the InfoSet to the target client and copies the assignment of the InfoSet to user groups from the source client, if corresponding user groups exist in the target client.

During an import, the import option UNASSIGN first deletes the assignment of that InfoSet to user groups in the target client. The InfoSet is then copied to the target client. The InfoSet is not assigned to user groups.



As described above, the import options REPLACE, GROUP=ug and UNASSIGN delete the assignment of that InfoSet to user groups in the target client first. If, however, a user group in the target client still has queries that refer to the InfoSet, the assignment remains.

Transporting a Query

When transporting a query, only the query itself is transported.

If the checkbox *Transport query variants* is selected when you perform the transport action *Export*, all the query variants, i.e. all the variants of the associated query report are also transported to the transport dataset.

To import a query, an appropriate user group and an appropriate InfoSet must exist in the target client, and the InfoSet must be assigned to the user group.



The query has the name G1 and belongs to the source client for the user group FB.

During an import, the import option REPLACE overwrites the query G1 of the user group FB in the target client, if this user group exists.

During an import, the import option GROUP=FX overwrites the query G1 of the user group FX in the target client, if this user group exists. Through this import option, the query is explicitly assigned to a user group in the target client. The user group can be different from the user group in the source client.

In both cases, you must ensure that a suitable InfoSet exists in the target system and that this InfoSet is assigned to the user group FB or FX.



When transporting a query, the system does not transport the generated report at the same time. After the transport, it ensures that the report is regenerated in the target system as soon as the query is executed the first time. This means that variants can be transported, although the associated report is not transported.

Reading Transport Datasets

Reading Transport Datasets

To read in a transport dataset, select one of the transport actions *Import* or *Upload* on the initial screen of the transport tool.

If you want to perform a test import first, select *Test run*. The system then carries out all the usual checks for an import and prepares an import log.

The import log also contains information on which locks need to be set during import. The test import enables you to check that importing a particular transport dataset will be successful and shows the changes that would be made in table AQLDB of the target client. Test runs never actual perform these changes themselves, however. You should always make use of this checking facility.

To perform an import (or test import), you must define the transport dataset(s) you want to import. The transport datasets in the table AQTDB have a key which matches the name of the transport request created for the transport system during the export. In the select option *Transport request*, you must specify the key of the transport datasets to be imported. The section [Managing Transport Datasets \[Seite 318\]](#) below describes how you can get an overview of the transport datasets in the table AQTDB.

If, for example, you want to re-import the dataset exported in [Generating Transport Datasets \[Seite 310\]](#), you must make the following entries:

Transport action	<input type="radio"/> Export	<input type="radio"/> Download	<input type="radio"/> Display
	<input checked="" type="radio"/> Import	<input type="radio"/> Upload	<input type="radio"/> Delete
	<input type="radio"/> Copy Standard Area -> Global Area		
	<input type="radio"/> Copy Global Area -> Standard Area		
<input checked="" type="checkbox"/> Test run	<input type="checkbox"/> Overwriting allowed (only with import/upload/copy)		
<input type="checkbox"/> Transport query variants (only with export/import/copy)			
<input type="checkbox"/> Transport the report-reprot interface for queries (only with export/import)			
<hr/>			
<input checked="" type="radio"/> Transport user groups	Import option	REPLACE	
User groups		to	
<hr/>			
<input type="radio"/> Transport functional areas	Import option	REPLACE	
Functional areas		to	
<hr/>			
<input type="radio"/> Transport functional areas and queries	Import option	REPLACE	
Functional areas		to	
Queries		to	
	Import option queries	REPLACE	
<hr/>			
<input type="radio"/> Transport queries	Import option	REPLACE	
User groups		to	
Queries		to	
<hr/>			
Dataset with imports	B20K901245	to	
<input type="checkbox"/> Delete after successful import			

When the import is finished, the system outputs a detailed log of all the checks performed and all the changes made to the table AQLDB.

If you have selected the *Delete after successful import* parameter, the system deletes the transport dataset from the table AQTDB after importing it, if no error occurs. You should use this parameter only if you are sure that you no longer need the transport dataset. The following section describes other ways you can delete transport datasets.

Managing Transport Datasets

You can use the functions *Display* and *Delete* in the transport tool to manage transport datasets in table AQTDB.

Whenever you choose the transport option *Display*, all transport datasets contained in transport table AQTDB are displayed with their name (which is the same as the name of the transport request) and contents.

If you want to delete the transport datasets, choose the transport action *Delete* and fill out the select option *Transport request*. All transport datasets are then deleted according to this select option.



In order to avoid unintentional deletions, you cannot *Delete* unless a value has been entered for the select option *Transport request*.

A log is generated containing a list of those transport datasets that have been deleted.

Transporting Objects Between Query Areas

Copying objects from one Query area to another is similar to transporting objects between different clients in a single system. Thus, you should go about copying objects between Query areas in much the same way as you would transport objects within the standard area. In principle, all objects may be copied to the other different Query area.

There are two transport options available on the initial screen of the Query transport tool that allow you to do this.

- *Copy Global Area → Standard Area*
- *Copy Standard Area → Global Area*

Both of these options are combinations made up of an export step and an import step. The only difference being that no transport dataset is created in transport table AQTDB and no transport request is generated in the Change and Transport Organizer. During import all of the usual checks are performed and a log is generated.

Copy Global Area → Standard Area copies Query objects from the global Query area to the standard Query area. This corresponds to object export out of the global area and object import into the standard area. The options *Test run*, *Overwriting allowed*, and *Transport Query variants* perform the same functions as described in [Generating Transport Datasets \[Seite 310\]](#). You can choose which objects you want to transport by entering their names in the input fields on the transport maintenance screen.

The development classes of those objects copied from the global area are ignored, that is their copies in the standard area have no development class or are local objects.

Copy Standard Area → Global Area copies Query objects from the standard Query area to the global Query area. This corresponds to object export out of the standard area and object import into the global area.

Pay attention to the following concerns related to development classes of objects copied into the global area:

- If an object in the global area is overwritten, the new object copied in inherits the development class of the object overwritten.
- If the object being copied into the global area is a new object, a window appears asking you to assign the object a development class.
Depending on what kind of development class you assign, it may or may not be necessary to assign a correction request as well. If the object being copied into the global area is a new Query whose assigned InfoSet or user group has a temporary development class, then the Query will automatically be assigned to development class \$TMP. Whenever this happens, no dialog box asking you to assign a development class will appear.

Language Comparison

Previous sections have described how to generate queries as well as how to create InfoSets and user groups. These operations involve dealing with a number of different text elements. For example, you have to specify descriptions for a field group when setting up an InfoSet. The system proposes descriptions for fields, but you can modify these if you wish. When defining a query, you must specify a title and you can also enter notes as well as your own headers for the different sub-lists and selection texts. You enter all these text elements in your own language, i.e. the logon language.

If, however, another user logs on in a different language and wants to use the queries and InfoSets you have defined, all the text elements should preferably be displayed in that language.

SAP Query provides a language comparison for this reason. The language comparison allows you to enter all relevant texts in one or more languages for a query or InfoSet. You can then use that query or InfoSet in various logon languages.

SAP Query notes the language variants that exist for each text and checks that the individual variants match. If two language variants reveal differences, the system also notes which of the two variants contains changes and indicates which variant has the most up-to-date text.

The language comparison facility is not intended to be used constantly. A language comparison should first take place when the query or InfoSet is stable, meaning that no more essential changes are expected. You can of course still carry out changes at any time. If you change texts that have already been matched, you must then perform another comparison. SAP Query provides utilities to help you analyze these texts.

You perform language comparison with the component *Language Comparison*:

The following sections explain which texts you compare individually. It is important that non-specific references exist. To be able to edit a query in different languages, you must first perform a language comparison for the relevant InfoSet, since many InfoSet texts are either used as an aid when creating the query or are copied directly to the query.

This chapter contains the following sections:

[General Language Comparison Procedure \[Seite 321\]](#)

[Language Comparison of an InfoSet \[Seite 325\]](#)

[Language Comparison of a Query \[Seite 327\]](#)

[Language Comparison for User Groups \[Seite 330\]](#)

General Language Comparison Procedure

You can use the component *Language Comparison* to compare the language texts of the different SAP Query objects. The queries and the InfoSets, and also the user groups, count as these objects. Texts are contained in all of these objects. In principle, the language comparison runs in the same way for all objects. This section thus describes the general principles.

When you call the *Utilities* → *Translation* → *SAP Query* component, you see the following initial screen:

The screenshot shows a web-based interface for language comparison. At the top, under 'Lang. comparison', there are two rows: 'From' with a dropdown set to 'EN' and 'English', and 'To' with a dropdown set to '...' and 'German'. Below this, there are three radio buttons: 'Query' (selected), 'InfoSet', and 'User groups'. To the right of the 'Query' radio button is a yellow input field with a magnifying glass icon. Below the 'Query' radio button is a label 'User group' followed by a yellow input field. To the right of the 'InfoSet' radio button is a yellow input field. At the bottom, there are two buttons: 'Change' (with a pencil icon) and 'Display' (with a magnifying glass icon).

You must first specify the source and target languages. The source language will normally be the language in which the object was created or last changed. In this way, you ensure that the texts in the source language are always the latest versions.

You specify the source and target languages with identifiers, for example 'DE' for German and 'EN' for English. These identifiers must be different and supported by the system. For a list of supported languages, display the possible entries for the language fields.

You then specify whether you want to compare the texts of a query, an InfoSet or a user group. To do this, select the appropriate radio button on the initial screen. With queries and InfoSets, you must also specify which query or InfoSet area you want to process.



You can display a list of queries or InfoSets using the input help for the individual input fields. It is possible to select a query, or an InfoSet, from this list.

As an example for this section, we can use the query G6 of the user group FB. You can start the language comparison by choosing *Change*. You then see a screen which contains a list of the texts used in this query.

General Language Comparison Procedure

	Sub-object	Description
<input checked="" type="checkbox"/>	Title and notes	Flight connections (1st example)
<input type="checkbox"/>	Field headers SPFLI-CARRID	Airline carrier Id
<input checked="" type="checkbox"/>	Basic list Basic list headers	

You then see an overview screen which lists (in the column *Sub-objects*) all parts of the query definition where texts are used. The sub-objects are classified in text groups and even into sub-groups if several sub-objects of the same type may exist. The column *Description* column contains a more precise explanation of each sub-object.

The first column on the overview screen indicates whether the texts of a sub-object have been compared or not. If this column is flagged, compared texts exist. Therefore, you can easily tell which texts remain to be compared.



A text is considered as compared if the component *Language Comparison* has been used to generate the source and target languages and if no changes have been made to any of the texts by another SAP Query component.

You can also restrict the overview screen to the sub-objects which have not yet been compared. To do this, select *Only differences*. The overview now only displays those sub-objects whose texts still must be compared.

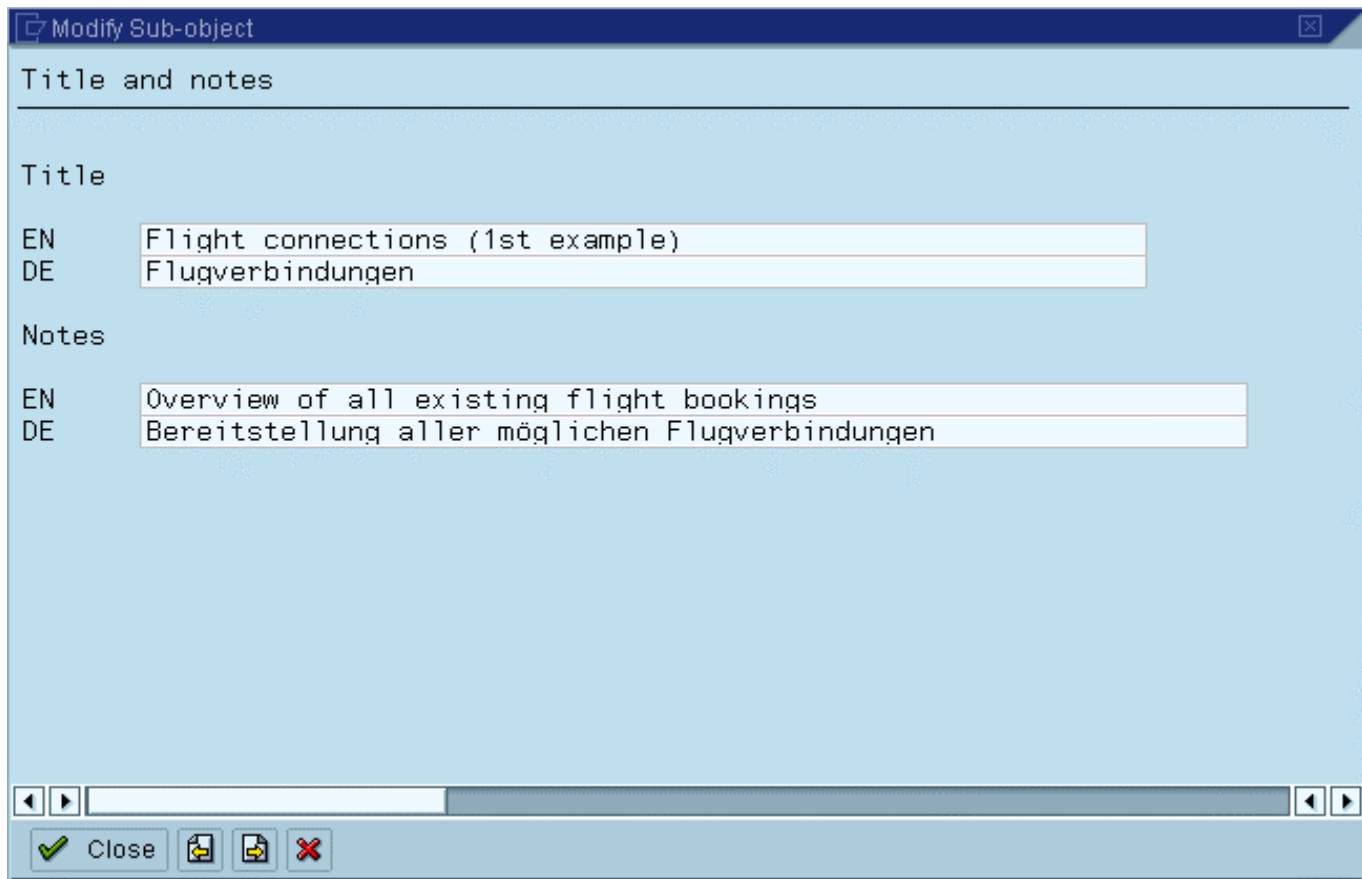


If the *Only differences* function concludes with the message that there are no more differences, all the texts have been compared. You can therefore use this function to check whether a language comparison is necessary at all.

If you want to return to the full overview screen containing all sub-objects, select *All texts*.

To compare the texts of a sub-object, place the cursor on the sub-object and select *Choose*. You then see a window with all the texts of the selected sub-object.

General Language Comparison Procedure



The window contains details about the sub-object and also the individual texts. Each text exists in two versions, the source language version and the target language version. Since the text fields are ready for input, you can make any changes you like. If the text version in one language is missing, the input field is blank.

If two texts have not been compared, one of the texts is flagged. This is either the text which is still missing or the "older" text. It is thus easy to tell which text needs to be changed within the context of the language comparison.

Now you can either change the texts found in the system or create totally new ones. You can change or expand the either one or both existing text versions.



The component *Language Comparison* does not allow deletion of texts. If you attempt to do this by overwriting a text with blanks, the system re-inserts the text you have just overwritten.

To return to the overview screen, select *Close*. If you have compared all the texts of the sub-object, it is now flagged as compared. However, you can still call the preceding or following sub-object in the list by using the functions *Previous sub-object* and *Next sub-object*.

Before returning to the initial screen, you must save any changes (with the *Save* function). You do receive a warning from the system, however, if you have forgotten to save. You can then carry out a save.

Language Comparison of an InfoSet

To compare the language texts of an InfoSet, specify the source and target languages on the initial screen, select the radio button for InfoSets and enter the InfoSet name in the appropriate field. As an example, we can use the InfoSet **FLBU** (flight bookings).

Choose *Change* to start the language comparison. You then see the following overview screen.

	Sub-object	Description
<input checked="" type="checkbox"/>	Functional area	Flight bookings
	Functional group	
<input checked="" type="checkbox"/>	10	Flight connections
<input type="checkbox"/>	20	Flights
<input type="checkbox"/>	30	Bookings
	Database table	
<input type="checkbox"/>	SBOOK	
<input type="checkbox"/>	SFLIGHT	
<input type="checkbox"/>	SPFLI	
	Database field	
<input checked="" type="checkbox"/>	SBOOK-BOOKID	Booking number
<input checked="" type="checkbox"/>	SBOOK-CLASS	Flight class
<input checked="" type="checkbox"/>	SBOOK-CUSTOMID	Customer number
<input checked="" type="checkbox"/>	SBOOK-CUSTTYPE	Business/private customer
<input checked="" type="checkbox"/>	SBOOK-FORCURAM	Booking price in foreign currency
<input checked="" type="checkbox"/>	SBOOK-INVOICE	Invoice flag (yes/no)
<input checked="" type="checkbox"/>	SBOOK-LOCCURAM	Price of booking in local currency
<input checked="" type="checkbox"/>	SBOOK-LUGGWEIGHT	Weight of luggage
<input checked="" type="checkbox"/>	SBOOK-ORDER_DATE	Booking date
<input checked="" type="checkbox"/>	SBOOK-SMOKER	Smoker
<input checked="" type="checkbox"/>	SFLIGHT-FLDATE	Flight date
<input checked="" type="checkbox"/>	SFLIGHT-PAYMENTSUM	Total of current bookings
<input checked="" type="checkbox"/>	SFLIGHT-PLANETYPE	Plane type
<input checked="" type="checkbox"/>	SFLIGHT-PRICE	Ticket price
<input checked="" type="checkbox"/>	SFLIGHT-SEATSMAX	Maximum capacity
<input checked="" type="checkbox"/>	SFLIGHT-SEATSOCC	Occupied seats
<input checked="" type="checkbox"/>	SPFLI-AIRPFROM	Departure airport
<input checked="" type="checkbox"/>	SPFLI-AIRPTO	Destination airport
<input checked="" type="checkbox"/>	SPFLI-ARRTIME	Arrival time

Subsequent procedure is exactly the same as described in the previous section. Here, the list contains only the different sub-objects which can occur with InfoSets, which texts are assigned to the sub-objects and what (if anything) needs to be noted for the sub-objects. The sub-objects include:

- InfoSet name
- Field group name
- Database tables
- Database fields

Texts of database fields exist in two forms - the description (or long text) and the column header.

Language Comparison of an InfoSet

Database field		SPFLI-CITYFROM
Description		
EN	From city	
DE	Startort	
Header		
EN	From city	
DE	Startort	

- **Table fields**
Table fields in this context are fields of linked additional tables included in a field group. As with database fields, the texts of table fields exist in two forms - the description and the header.
- **Additional fields**
Additional field texts exist in two forms - descriptions and headers.
- **Parameter**
There is a description for each parameter. If a special text has been specified for the selection screen when defining the InfoSet, this selection text is also offered for comparison.
- **Selection criteria**
As with parameters, there are descriptions and, in some cases, texts for the selection screen.

In InfoSets, you can reduce the translation effort considerably by choosing *Edit → Get standard texts*. This function determines - for database tables, database fields and table fields - the existing target language texts in the Dictionary and copies them to the InfoSet, provided they are not already present. If a source language text in the InfoSet still matches the corresponding Dictionary text, the texts for the relevant sub-object are flagged as modified.

Language Comparison of a Query

To compare the language texts of a query, specify the source and target languages on the initial screen, select the radio button for queries and enter the query and user group names in the appropriate fields. As an example, we can use the queries G6 and S3 of the user group FB.

Choose *Change* to start the comparison. You then see the following overview screen.

	Sub-object	Description
<input checked="" type="checkbox"/>	Title and notes	Quarterly sales per airline for one year
	Local field	
<input checked="" type="checkbox"/>	Q1/%Z_0002	Bookings for the 1st quarter
<input checked="" type="checkbox"/>	Q2/%Z_0005	Bookings for the 2nd quarter
<input checked="" type="checkbox"/>	Q3/%Z_0006	Bookings for the 3rd quarter
<input checked="" type="checkbox"/>	Q4/%Z_0007	Bookings for the 4th quarter
<input checked="" type="checkbox"/>	Y/%Z_0001	Year
	Field headers	
<input checked="" type="checkbox"/>	SPFLI-CARRID	Airline carrier Id
	Statistics	
<input checked="" type="checkbox"/>	01	Quarterly revenues

Since the subsequent procedure is already known, the list here contains only the different sub-objects or texts which can occur, what (if anything) needs to be noted.

- Title and notes
- Local fields
 - Texts of local fields exist in two forms - a description (or long text) and headers. Please note that, with headers, replacement variables for field contents (**&field**) occur in every language version.

Language Comparison of a Query

Modify Sub-object

Local field Q1/%Z_0002

Description

EN	Bookings for the 1st quarter
DE	Buchungen im 1. Quartal

Header

EN	1st quarter &Y
DE	1. Quartal &Y

Close

- **Field headers**
Field headers are always offered for comparison if non-standard headers have been defined in the query definition. As with local fields, replacement variables for field contents (**&field**) occur in every language version.
- **Selection texts**
These are texts which identify the declared additional selections on the selection screen.
- **Control level texts**
If you have defined control level texts for sort criteria and these texts are to be output at the beginning of each control level, they are also offered for comparison. Please note here that the template in each control level text for the value of the control level field must occur in every language version. It is especially important that the delimiters for the template (< and >) and the number of underscores are retained.
- **Sub-total texts**
The same applies for sub-total texts, i.e. the texts which are output at the end of each control level if a sub-total has been requested, as for control level texts. The template for the value of the control level field must be retained in every language version.
- **Field templates**
Field output templates also need to be compared if they contain texts. However, it is very important that the delimiters (**LL<** and **>**) and the number of underscores are retained in every language version.
- **Basic list headers**
The fixed header and footer lines defined for a basic list can be compared. Please also note here that replacement variables for field contents (**&field**) occur in every language version.

- Statistics

The title, the fixed header lines and the footer lines must be compared for each statistic.
As with basic list headers, replacement variables for field contents (**&field**) occur in every language version.
- Ranked lists

The same texts exist for ranked lists as for statistics and the language comparison procedure is similar.

Language Comparison for User Groups

Language Comparison for User Groups

The only texts generated with user groups are the user group long texts. These texts are also processed by the language comparison facility, i.e. there should be one long text for each user group.

To compare the language texts of user groups, specify the source and target languages on the initial screen and select the radio button for user groups.

Choose *Change* to start the language comparison.

Appendix

Overview of SAP Query Functions

Overview of SAP Query Functions

The SAP Query components contain a number of important and similar functions. In the list below, you can see the assignment of these functions to function keys. This is the same for all components and conforms to SAP standards.

F1:	Help
F2:	Choose
F3:	Back
F4:	Possible entries
F5:	Create; Previous screen
F6:	Change, Next screen
F7:	Description
F8:	Execute
F9:	Copy
F11:	Save
F12:	Cancel
F13:	Print, Layout display, Transports
F14:	Delete
F15:	Exit
F16:	Basic list, Generate
F17:	Statistics
F18:	Ranked list
F19:	Display help texts
F20:	Without help texts, Full screen
F21:	First page
F22:	Previous page
F23:	Next page
F24:	Last page

For a detailed explanation of the individual functions, refer to the online documentation, if the meaning is not immediately clear from the function name. There are, of course, other component-specific functions assigned to the function keys in the various SAP Query components. You can display the actual key setting on each screen.

Queries

A query definition generates an ABAP report with the name 'Aqmmb...bq q' where

- **mm** stands for the client in the standard area
For the clients 0 to 99, mm = 00 - 99.
For clients > 99, mm = a two-character key determined by the system. In the global area mm=ZZ
- **b...b** stands for your user group
- **q...q** stands for the query name
- Blank characters in names (for example when a user group contains less than 12 characters) are replaced by equals signs ('=').



Using queries as a foundation for your own developments is not recommended. Every query has a runtime system assigned to it whose interfaces can change from release to release. Therefore it is in no way guaranteed that queries developed 'by hand' can still be run after a new release has been implemented.

Enhancement SQUE0001: Private File

Enhancement SQUE0001: Private File

SAP Query has several options which allow you to pass on data generated in queries to other software products for further processing. These options include:

- Table calculation program (for example EXCEL) via the XXL interface
- SAP Business Graphics
- Executive Information System (EIS)
- Download to file

In each case, the procedure is the same. The generated data is placed in an internal table (the data table), where the field sequence and type corresponds to that in the query list. Another internal table (the description table) contains a description of the individual fields in the data table. Apart from the type and position in the list, the table also contains information about the field names. Both these tables are passed to the above-mentioned software products for further processing.

Through enhancement SQUE0001, SAP Query also allows you to use the same interface to access other software products. SQUE0001 consists of the following components:



The *Private file* function has largely been made obsolete with the introduction of the *Additional function pool* (see [Additional Function Pool \[Seite 130\]](#)). SAP will continue to support the *Private file* function for reasons of compatibility. It is, however, recommended that you gradually re-define those functions that you have in the *Private file* so that you can add them to the *Additional function pool*.

Be aware that additional function pool functions can only be processed online in the foreground and thus CANNOT be called during background processing. In contrast, contents of the private file CAN be called during background processing.

SQUE0001 consists of the following two components:

- the function module EXIT_RSAQEXCE_001
- the menu enhancement RSAQEXCE+DAT

When you activate the enhancement in a project (using Transaction CMOD), an additional interactive function called *Private file* (function code: + DAT) is made available in each report generated by the SAP query, which calls the aforementioned function module when activated. The tables mentioned above (data table and description table) are then passed to the function module. The selection screen thus contains an additional parameter which, like the other interactive functions (EXCEL, Graphics, EIS, Download to file and Save), allows the user to execute the query without displaying the list beforehand. Since the name of this parameter on the selection screen is *Private file*, you are recommended to give the menu enhancement the same name. This corresponds to the SAP standard.



When you have activated the enhancement, you should regenerate all queries, so that the additional parameter appears on the selection screen. You can achieve this best by going into the component *Maintain InfoSets* and selecting *Goto → Query directory*.

Function module EXIT_RSAQEXCE_001

This section describes which parameters are passed to the function module EXIT_RSAQEXCE_001 and how the values they contain are interpreted or processed. The following example demonstrates this process.

In client 03, a user in user group RX has created a query X1 which consists of a one-line basic list, a statistic and a ranked list. The basic list outputs the following fields in order:

KNA1-KUNNR	Customer number
KNA1-NAME1	Name
KNC1-GJAHR	Fiscal year
KNC1-UM01U	Sales in posting period 1
T001-WAERS	Currency key

. The statistic outputs the following fields in order:

KNA1-LAND1	Country key
KNC1-UM01U	Sales in posting period 1

For the field KNC1-UM01U, details are also required about the number of records read, the percentage value and the average value. The ranked list outputs the following fields in order:

KNA1-KUNNR	Customer number
KNA1-NAME1	Name
KNC1-UM01U	Sales in posting period 1

Parameters of function module EXIT_RSAQEXCE_001

The function module EXIT_RSAQEXCE_001 has the following parameters:

PROGRAM	The program which passes the parameters
LIST_ID	The ID of the passed sub-list
LISTTEXT	The title of the passed sub-list
DATATAB	The data table
LISTDESC	The description table for the fields of the data table

PROGRAM parameter

The PROGRAM parameter contains the name of the query report which called the function module. From this name, you can determine which query has been processed, since the structure of the name is always as follows:

Aqmmmb...bq...q

cc	= client or ZZ in the global area
b...b	= user group name
q...q	= query name

In terms of our example, the name of the report is therefore

AQRX=====X1=====.

LIST_ID parameter

The LIST_ID parameter contains a three-character ID which specifies the data table passed. Each list generated by a query can comprise several sub-lists (a basic list, several statistics and

Enhancement SQUE0001: Private File

several ranked lists). The data table (DATATAB) transferred to the function module contains only the data for one sub-list respectively. The description table (LISTDESC), however, always contains the description for the fields of all sublists.

The possible parameter values are

G00	if DATATAB contains the data of the basic list
Txx	if DATATAB contains the data of the statistic xx (01,02,...)
Rxx	if DATATAB contains the data of the ranked list xx (01,02,...)

The **LIST_ID** parameter thus has two functions. Firstly, it allows you to determine the sub-list type involved. Secondly, you can define the relevant entries in the description table.

In the current example, if the data of the basic list is transferred, the parameter **LIST_ID** contains the value **G00**. If the statistics data is transferred then the value is **T01**. If the ranked list data is transferred then the parameter contains the value **R01**.

LISTTEXT parameter

The **LISTTEXT** parameter contains a text which is assigned to the relevant list when you define the query.

DATATAB parameter

The **DATATAB** table contains the data of the sub-list passed. The sequence and type of the fields correspond to the sequence of the fields output in the sub-list. One table line corresponds to one list line respectively. Any existing line breaks in basic lists are ignored here (that is they always form one table line).

Since the structure of the data table depends on how you define the sub-list to be passed, you have to define this structure beforehand in the function module. For this purpose, you can use the description table **LISTDESC**. To access individual fields in the data table, you use field symbols and the **ASSIGN COMPONENT** statement.

If you pass the basic list in our example, one line of the data table consists of 5 fields with the following types, lengths and number of decimal places:

1. Field: Type C, Length 10	
2. :	(-)
3. Field: Type N, Length 04	(-)
4. Field: Type P, Length 08, Dec. 02	(-)
5. :	(-)

If you pass the statistic, one line of the data table consists of 5 fields with the following types, length and decimal places:

1.	-
2. .	
3. Field: Type I, Length 04	
4. ,.	
5. .	

If you pass the ranked list, one line of the data table consists of 4 fields with the following types, lengths and decimal places:

1.	
2.	
3.	
4.	

To define the type, length and number of decimal places, you use either the ABAP statement DESCRIBE FIELD or the description table LISTDESC (see below).

LISTDESC parameter

The LISTDESC description table contains a description of the fields in the data table DATATAB. This includes not only the type, length and number of decimal places, but also any information you cannot define with the DESCRIBE FIELD statement. If a query comprises several sub-lists (as in our example), the description table always contains a description of the fields in all data tables (that is all sub-lists) which can be passed by this query. You can use the LIST_ID parameter to retrieve the appropriate entries from the description table.

In our example then, the description table always contains the descriptions of fields of the data tables for the basic list, the statistics and the ranked list.

The description table has the structure RSAQLDESC. This structure is made up of the following fields:

FNAMENew	type C, length 70	field name (source field) (Rel.4.0)
FNAMEINT	type C, length 30	field name (data table)
FKEY	type C, length 1	flag for lead column
FGTYPE	type C, length 1	flag for graphic type
FNZERO	type C, length 1	flag for zero display
FSUM	type C, length 1	summation flag
FSONLY	type C, length 1	sorting flag
FSORT	type N, length 5	sorting level
FSODS	type C, length 1	sorting direction
FSUBT	type C, length 1	subtotal flag
FLPOS	type N, length 4	position of field (like FPOS)
FNUMB	type C, length 2	Field group ID
FNUMBT	type C, length 24	Field group text
LID	type C, length 03	ID of the sub-list
FPOS	type N, length 02	position of field
FCONT	type N, length 01	number of continuation line
FCURPOS	type N, length 03	list position of field
FGRAF	type C, length 01	text flag for graphic
FTYP	type C, length 01	ABAP type of field

Enhancement SQUE0001: Private File

FLEN	type N, length 03	length of field
FDEC	type N, length 02	number of decimal places for field
FOLEN	type N, length 03	output length of field
FCUR	type C, length 01	currency flag
FADD	type C, length 01	flag for internal field
FDESC	type C, length 40	field description (long text)
FNAME	type C, length 30	field name (source field)
FCOL	type C, length 30	column header for field

LID	indicates to which query sub-list the field belongs. The parameter values can be the same as for the LIST_ID parameter (see above)
FPOS	describes the position of the field in the sub-list specified by the LID parameter. The positions are counted consecutively from 1 (for example 01, 02,...).
FCONT	describes the number of the continuation line in which the field is output (0,1,...). This can occur with basic lists that collapse due to a too small list width (LINE SIZE). Field numbering (as in FPOS) is not affected, since it is independent of the FCONT value. Normally, this field is not required for the <i>Private file</i> function.
FCURPOS	describes the start position of the field in the list line, if the field is numeric (FTYP = P, I, F). Otherwise, FCURPOS contains the value 000. Normally, this field is not required for the <i>Private file</i> function.
FGRAF	contains the value X, if you use the value of the non-numeric field as a help text for the graphic. Normally, this field is not required for the <i>Private file</i> function.
FTYP	describes the ABAP type of the field and can accept the values C, D, F, I, N, P, T, or X.
FLEN	describes the length of the field in bytes.
FDEC	describes the number of decimal places for the field. This specification is only significant if FTYP contains the value P. In all other cases, FDEC contains the value 00.
FOLEN	describes the output length of the field in the sub-list, as specified when defining the query.
FCUR	identifies the field as F currency amount field W currency key or currency field M quantity field E unit field

FADD	indicates whether the field is an additional field generated by the query itself. The following values are allowed: R rank specification (only in ranked lists) C number of records read (only in statistics) P percentage specification (only in statistics) M average value (only in statistics)
FDESC	contains the field description which the user sees when defining the query.
FNAME	contains the field name from the query report from which the information for the data table is derived (up to Release 4.0)
FCOL	contains the one line column header for the data table field described by an entry in the LISTDESC table.
FNAMENEW	like FNAME (from Release 4.0)
FNAMEINT	contains the field name from the data table. Use this name together with the statement ASSIGN COMPONENT to dynamically access the field in the data table.
FKEY	contains the value X if the field has been output as a key field (lead column) in the list.
FGTYPE	identifies the field's contents as S a symbol I an icon
FNZERO	contains the value X if the field has been output with the option NO-ZERO in the list.
FSUM	contains the value X if the field has been totaled in the list.
FSONLY	contains the value X if the list has been sorted according to this field, but the field itself was not output.
FSORT	contains the sorting number if the list has been sorted according to the field.
FSODS	contains the value X if the list has been sorted in descending order according to the field.
FSUBT	contains the value X if the list has been sorted according to the field and sub-totals were generated during sorting.
FLPOS	like FPOS
FNUMB	field group of the field whose name is stored in FNAMENEW.
FNUMBT	Long text of the field group

Example**Example**

In our example, the table LISTDESC would have the following structure (the values of the FDESC fields are also listed).

	FDESC	FNAME/FNAMENEW	FCOL	...
G00 01 0 000 X C 010 00 010	1)	KNA1-KUNNR	Number	
G00 02 0 000 X C 035 00 035	2)	KNA1-NAME1	Name	
G00 03 0 000 X N 004 00 004	3)	KNC1-GYEAR	Year	
G00 04 0 053 P 008 02 021 F	4)	KNC1-UM01U	Sales	
G00 05 0 000 X C 005 00 005 W	5)	T001-WAERS	Currency	
T01 01 0 000 X C 003 00 003	6)	T01-KNA1-LAND1	Land	
T01 02 0 005 P 016 02 021 F	7)	T01-KNC1-UM01U	Sales	
T01 03 0 027 I 004 00 008 C	8)	T01-C\$T-KNC1-UM01U	Total No.	
T01 04 0 036 P 004 03 009 P	9)	T01-P\$R-KNC1-UM01U	%	
T01 05 0 047 P 008 02 021 F M	10)	T01-A\$G-KNC1-UM01U	Average	
R01 01 0 000 C 006 00 008 R	11)	V01-RANGCT	Rank	
R01 02 0 000 X C 010 00 010	12)	V01-KNA1-KUNNR	Number	
R01 03 0 000 X C 035 00 035	13)	V01-KNA1-NAME1	Name	
R01 04 0 057 P 016 02 021 F	14)	V01-KNC1-UM01U	Sales	

1) Customer number
 2) Name
 3) Fiscal year
 4) Sales in posting period 1
 5) Currency key
 6) Country key
 7) Sales in posting period 1
 8) Total amount
 9) Share in %
 10) Mean value
 11) Rank
 12) Customer number
 13) Name
 14) Sales in posting period 1

When you call the function module, the entries of significance in the table LISTDESC are all those where LISTDESC-LID = LIST_ID.

Example: Implementing EXIT_RSAQEXCE_001

In conclusion, a simple example demonstrates the implementation of the function module EXIT_RSAQEXCE_001. Here, the passed tables LISTDESC und DATATAB appear as a list and, in the case of the table LISTDESC, the fields FCONT, FCURPOS and FGRAF are omitted. The example also clearly shows how to interpret the individual parameters and how to access the data table.

```
***INCLUDE ZXQUEUE01.
```

```
FIELD-SYMBOLS <F>.
```

```
NEW-PAGE LINE-SIZE 132.
```

```
WRITE: / 'Query', PROGRAM+16(14), 'of user group', PROGRAM+4(12).
```

```
ULINE. SKIP 1.
```

```
WRITE: / 'Fields of sub-list', LIST_ID.
```

```
ULINE.
```

```
LOOP AT LISTDESC WHERE LID = LIST_ID.
```

```
  WRITE: / LISTDESC-FDESC,  
         LISTDESC-FNAME,  
         LISTDESC-FPOS,
```

```

LISTDESC-FTYP,
LISTDESC-FLEN,
LISTDESC-FDEC,
LISTDESC-FOLEN,
LISTDESC-FCUR,
LISTDESC-FADD.
LISTDESC-FCOL.
ENDLOOP.
SKIP 1.
WRITE: / 'Data table'.
ULINE.
LOOP AT DATATAB.
  NEW-LINE.
  DO.
    ASSIGN COMPONENT SY-INDEX OF STRUCTURE DATATAB TO <F>.
    IF SY-SUBRC <> 0. EXIT. ENDIF.
    WRITE <F>.
  ENDDO.
ENDLOOP.
ULINE.

```

Access Optimization in Queries

You can get considerably improved performance from ABAP reports thanks to enhancements made in OPEN SQL. SAP Query exploits these enhancements in order to achieve an improved runtime for query reports.

One important enhancement is that it is now possible in all SELECT statements to specify a list of fields that should be read from the database table. If '*' is used for the field list then all the fields are read. Since time-consuming conversions are often applied to fields when they are read from the database into program fields, this procedure is particularly ineffective if the database table contains several fields of which just a few are really needed in the program. This is in fact the usual case for query reports.

The GET statement has been enhanced, which also affects field lists in SELECT statements. It is also possible to specify here a list of fields that are required in a report. Only these fields are read from the logical database. The connection between the GET and SELECT statements is that in a database program the GET statement is ultimately implemented using the SELECT statement. However, it is possible to specify a field list in a GET statement only if the logical database being used supports these.

The term access optimization refers to the way that Query supports full usage of field lists in SELECT and GET statements. The procedure is as follows:

When an InfoSet is generated a list is created containing all the fields that are needed when the InfoSet is used in a query (the reference list). Additional tables and sections of code (additional fields, code that is called at GET, GET LATE and record processing) are particularly significant here, since other fields are accessed at these points. The purpose of the reference list is to ensure that the report generator of the query can determine all the fields that are implicitly required as a result of the use of an additional table or code from the directly needed fields contained in the list. Given this information, the report generator is then able to specify field lists for all SELECT and GET statements that are generated.

If an InfoSet has no additional fields and no sections of coding, the reference list can be created completely and without errors when the InfoSet is generated. Problems could arise if you use ABAP code. It is then possible that the InfoSet does not contain all the information needed for determining the required fields, making the reference list incomplete. This can mean that when query reports are processed some of the fields needed only ever get given their initial values.

If an InfoSet contains sections of code that could cause problems, a warning is output each time that it is generated. A check should then be made on whether any of the cases described below has occurred and the InfoSet must be corrected as appropriate.

It is necessary to know which fields are required for generating query reports in order to be able to draw up the reference list. These can be the following fields:

- Fields that were included in field groups
- Fields that are used for formulating the WHERE condition of connected additional tables
- Fields that are accessed from code belonging to additional fields
- Fields that are accessed from code called at GET / GET LATE or at record processing

The freedom to use any ABAP language construct at any point in a section of code means that it is possible to access fields without explicitly referring to these fields (field symbols, external Perform, DO... VARYING, ADD... THEN... UNTIL, etc.). However, in order to ensure that query reports can be generated without errors, it is essential that all fields accessed can be determined in EVERY section of code (additional fields, GET / GET LATE / record processing). This means that every field used in every section of code has to be named explicitly. If a section of code contains ABAP statements that access certain fields implicitly, then the ABAP FIELDS statement

must be used in the code to ensure that all database fields, table fields and additional fields that are used are named explicitly.



The KNC1 table contains the UM01U, UM02U and UM03U fields. These fields contain the monthly sales figures for the first three months of a year. An additional field, Q1, is to be used for the sales figure for the first quarter. This field calculates the sum of the other three fields with an external Perform.

```
PERFORM QUARTAL1(pppppppp) USING Q1.
```

The KNC1-UM01U, KNC1-UM02U and KNC1-UM03U fields are accessed via the common memory for the KNC1 table in the query report and in the program pppppppp that gets called. However, it is not evident from the code fragment above that these fields are actually required. Therefore this code fragment needs to be modified as follows:

```
PERFORM QUARTAL1(pppppppp) USING Q1.
```

```
FIELDS: KNC1-UM01U, KNC1-UM02U, KNC1-UM03U.
```

It is important to note that all fields required must be explicitly named in every section of code. This is because each section of code is included in the query report only if it is really used, and therefore it has to be clear from individual section of code which fields are needed in that particular section of code.

Note also that only those fields that are directly used in a section of code need to be named there.



The additional fields F1 and F2 are defined in the following code:

```
F1:      F1 = TAB-FELD. "TAB-FELD is a database field
```

```
F2:      F2 = F1 + 2.
```

Even though F2 indirectly accesses TAB-FELD, it is not necessary to list TAB-FELD in the code of F2 as being one of the fields used. This kind of indirect reference is automatically resolved when the InfoSet is generated. The sections of code for both additional fields is therefore correct exactly as given above.

In exceptional cases the following statement can be used in code:

```
FIELDS TAB.
```

where TAB is a database table or an additional table. This has the effect of using all the fields of the TAB table in the query report. However, this means that the access to the TAB table is no longer optimized and that therefore a considerable loss of performance can be expected when processing queries.

If the advice given here on maintaining an InfoSet is carefully followed, it will always be possible to generate a complete reference list. However, the generation process itself cannot check whether the InfoSet has been maintained correctly, and therefore it is always possible for a faulty case to "slip through the net", resulting in an incomplete reference list being generated. If a reference list is incomplete for a query report, this could lead to fields that are needed not being read from the database and therefore always retaining their initial values. Therefore there is a special test available for queries.

If a query returns incorrect results and the suspected cause is an incomplete reference list, then a query report can be generated without access optimization. This is done by calling the *Extras* → *Optimization on/off* function on the *Title, Format* screen of the query maintenance component. After this function has been called the first time, a check box appears on the screen and is checked. This means that access optimization is no longer active for this query. All fields of all

Access Optimization in Queries

tables are then read. The access optimization can be reactivated either by calling the *Extras* → *Optimization on/off* function again or by deselecting the check box.

The access optimization should be deactivated only for test purposes. If the query works correctly when the access optimization is deactivated then the InfoSet must be investigated and corrected so that a complete reference list gets generated (see above). If the query does not work correctly even with the access optimization deactivated then the problem is being caused by something else.

Following a test without access optimization and any necessary processing of the InfoSet, you should always switch access optimization on again. If you do not do this you will see a considerable drop in performance.

InfoSets and Structural Changes in a Logical Database

There should be as few changes as possible made to a logical database. Those that are made should be upwards compatible. This means that tables may be extended to include new fields and that new tables may be incorporated into the tree in such a way that the existing structure does not get changed but forms a subset of the new structure.

When a new InfoSet is created over a logical database, the structure of the logical database (the arrangement of its tables in the tree) is read from the relevant system tables once and is stored in the InfoSet. Whenever relationships between tables have to be evaluated, for example when query reports are being generated, this structure is accessed. Each time that an InfoSet is generated or the InfoSet maintenance is called, the structure of the logical database is read again and is compared with the structure stored in the InfoSet. This means that an InfoSet can be adapted to changes in the structure at this point.

Changes to individual tables (for example new fields, deleted fields, changes to the technical properties of fields) are usually straightforward to handle, since they are easily recognized and evaluated. If necessary, the appropriate warnings and error messages are shown when the generation is taking place.

There are, however, rather more restrictions when it comes to adapting an InfoSet to a change in the structure of a logical database. As long as the change of structure is nothing more than the insertion of a new table in the tree, then this is also a straightforward adaptation that is performed without special messages being issued. But in the cases that are described in the following, a message is output and the InfoSet has to be checked and possibly revised

If a table is removed from the structure and the selected fields of this table have not been used in queries, then the table together with the selected fields are also removed from the InfoSet. It is, however, still important to make a check, since fields from this table may be accessed in sections of code or in WHERE conditions contained in connected additional tables. These references must also be removed.

In the case that the structure of tree is changed so that the relationships between individual tables change, then the sections of code and the WHERE conditions of the additional tables also have to be checked. It is particularly important to note that the access paths in the tree (the paths from the root node to the current node) will now have changed and that therefore the set of fields that can be accessed has also changed. These kinds of changes can be very critical. It is no longer possible to be sure that the queries over the InfoSet still work exactly as they used to. Therefore these must also be checked and revised where necessary.

If a table is removed from the structure and the fields of this table are in use in queries, then it is neither possible to generate nor to change the functional area. The InfoSet can no longer be used in its current form. The *Goto → Query directory* function on the initial screen of the InfoSet maintenance component should be called to determine which queries use the table that has been removed (note: this function has been extended to output the tables that are referenced by each query). These queries need either to be deleted or to be changed so that the table that has been removed is no longer referenced. The InfoSet can then be generated or changed again.

Logical Databases with Different Node Types

From Release 4.0A, you have the option of defining logical database nodes in various different ways.

1. Type T nodes (ABAP Dictionary tables); called type *Database tables* from 4.5A
This type of node corresponds to the only type of node that existed prior to Release 4.0A (a flat Dictionary structure).
2. Type S nodes (structure from the ABAP Dictionary), called *Dictionary Type* from 4.5A,
In Releases 4.0A and 4.0B, this node type makes it possible for a flat Dictionary structure to be provided with an alias name that can be chosen at will within the logical database. This is needed if the name of one of these structures is longer than the maximum length of 14 characters allowed for node names. In addition, this node type allows you to include a Dictionary structure in a logical database numerous times under different aliases.
From Release 4.5A you can define complex types, meaning nested structures and table types, in the Dictionary. This node type allows you to use nodes with this kind of complex Dictionary type.
3. Type C nodes (complex data objects); called type *Data type* from 4.5A
These types of nodes refer to an arbitrary number of complex types that are defined in a type pool. From Release 4.5A, types of this sort can be better represented using complex ABAP Dictionary types. It is therefore no longer recommended that you use them in logical databases, as this leads to certain disadvantages in ABAP Query.

When defining InfoSets in Releases 4.0A and 4.0B you may only use those logical databases containing type *Database table* (type T) nodes. From Release 4.5A logical databases containing both other types of nodes can also be used. There are, however, some restrictions to be taken into account.

When using nodes with complex types (*Dictionary type* or *Data type*), ABAP Query only allows you to evaluate those fields that can be addressed directly from the highest level of the type. If the complex type in question contains a table, the individual components that the table is made up of cannot be addressed. This is demonstrated in the following example.

Given: The following kind of complex type. ABAP Notation is chosen. This kind of type could also be depicted using a *Dictionary type*.

```

TYPES: BEGIN OF person,
        name(20),
        first_name(20),
        birthday(10),
      END OF person,
      BEGIN OF address,
        street(30),
        number(6),
        zipcode(10),
        city(30),
      END OF address,
      address_table TYPE address OCCURS 0.

TYPES: BEGIN OF personal_data,
        person TYPE person,
        first_adress TYPE address,
        other_addresses TYPE address_table,
      END OF personal_data.
```

If the logical database node PDATA is of type PERSONAL_DATA, the following fields of this node can be evaluated using ABAP Query:

PDATA-PERSON-NAME
 PDATA-PERSON-FIRST_NAME
 PDATA-PERSON-BIRTHDAY
 PDATA-FIRST_ADDRESS-STREET
 PDATA-FIRST_ADDRESS-NUMBER
 PDATA-FIRST_ADDRESS-ZIPCODE
 PDATA-FIRST_ADDRESS-CITY

Those fields containing information about additional addresses (OTHER_ADRESSES) cannot be evaluated because a table type is being used here.

Additional restrictions apply to nodes with type *Data type* (type C). Due to the fact that nodes of this type are not defined as ABAP Dictionary types, no long texts or headers exist for these fields. Therefore, when defining an InfoSet, you can only use the technical names of the fields as name suggestions (for example PERSON-NAME). In order for these fields to appear in the same way that all other database nodes do, you must maintain all texts for them.

Additionally, pay attention to the fact that reference field definition is not possible with nodes of type *Data type* since this is only allowed with Dictionary types. Because of this, ABAP Query cannot recognize values with units (currency sums or quantities), nor can it evaluate them accordingly within these nodes.

Interactive Multiple-Line Basic Lists

Up to Release 4.6A, you may only use SAP Query's interactive functions (EXCEL, table display, display as an interactive list, graphics, etc.) with single line basic lists, statistics, and ranked lists. From Release 4.6A, almost all multiple-line basic lists are interactive. This means that the system treats them as single line lists and allows you to use them in conjunction with all of the functions listed above. This also allows you to arrange direct interactions, that is to directly call one of these interactive functions without having to display the list first.

In order for multiple-line basic lists to be interactive, the data presented in the list must be entered into a flat internal table. This table's line structure is determined by attaching all fields of all lines from the basic list one to the next, one right after the other. Each time that a list line is output, the fields from that list line are transferred to their corresponding fields in the table line. New lines are only ever added to the internal tables when all fields within a table line are filled, or when it is apparent in the list that those fields still missing will not be displayed in the list.

If, for example, a basic list has been defined with two lines (line 1 and 2) and a random number of line 2s can belong to line 1, you must distinguish between the following cases:

- Exactly one line 2 belongs to every line 1 found in the list.
In this case, exactly one line is inserted in the internal table containing the values of both lines.
- Multiple line 2s belong to every line 1 found in the list.
In this case, multiple lines are added to the internal table; the fields found in line 1 repeat themselves and the fields from line 2 are new for each new table line.
- No line 2s whatsoever belong to every line 1 found in the list
In this case, exactly one line is added to the internal table; in it the fields for line 2 are initial.

If a basic list contains more than two lines, the process described above occurs at multiple levels.

In general, all basic lists that do not retrieve their data from a logical database are interactive. If a list's data comes from a logical database, all fields used in the list may lie on one path in the database. In other words, those multiple-line basic lists that process fields from parallel tables are not interactive.

Logical Database F1S

The logical database F1S is used in this user guide to demonstrate the options available to users in SAP Query. Although it is a very simple model, this logical database contains all the important aspects of logical databases.

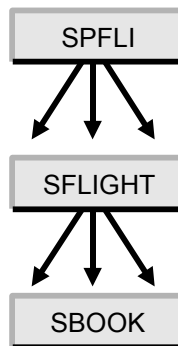
Logical database F1S contains data for flight bookings. The database contents are updated each time a booking is made. Any evaluation of this database therefore always reflects the current state of flight bookings.

Structure of the Logical Database

Logical database F1S contains three tables:

SPFLI	Information about individual flight connections
SFLIGHT	Information about individual flights for each flight connection
SBOOK	Information about bookings for a flight

These tables are arranged in a hierarchy as follows:



For each record (flight connection) in SPFLI, there are several records (flights for each flight connection) in SFLIGHT. For each record in SFLIGHT, the table SBOOK contains one record for each booking made.

Structure of Table SPFLI

The table SPFLI contains information about individual flight connections.

A flight connection is identified by the airline carrier (short name) and a flight number (code) which together form the table key. For each flight connection, the table contains information about departure and destination (city, airport, and time) distance and flight duration. A local currency is also specified for each flight connection. This is the currency which the airline carrier uses for its invoices.

Structure of Table SFLIGHT

The table SFLIGHT contains information about individual flights for each flight connection.

A flight is identified by the flight connection (airline carrier and flight number) and the flight date, which together form the table key. For each flight, the table contains information about the plane (type and number of seats), the price, and some other compressed data about bookings already made. The price of a flight is normally specified in the currency used at a flight connection's point of departure.

The data about bookings already made covers the number of bookings and the total amount taken. Since individual bookings can be made in different currencies, this total is always specified in the local currency of the airline carrier.

Logical Database F1S**Structure of Table SBOOK**

The table SBOOK contains information about individual bookings.

A booking is identified by the flight connection (airline carrier and flight number), the flight date, and a booking number, which together form the table key. For each booking, the table contains information about the customer (customer number, business or private customer, with or without invoice creation), the booked seat (class, smoker), the amount paid, and the weight of the luggage.

The amount paid for the booking is specified in two currencies - first in the currency in which payment was actually made and second in the local currency of the airline carrier. Discounts that can be negotiated when a booking is made can mean that the amount paid can be less than the fare specified in the table SFLIGHT.

Tables with Additional Information

The tables of logical database F1S also contain a range of information only in short form. Only short names are given for the airline carriers, for example LH for Lufthansa. Detailed information about an airline carrier (for example its full name) is stored in a table that is not part of the logical database.

The reason for this is as follows. The airline carrier is not the only key in table SPFLI. It occurs several times in this table - once for each flight connection it runs. Saving more detailed information in table SPFLI would be largely redundant. Only space-saving short codes are therefore used in table SPFLI. Whenever detailed information is needed, you can retrieve it by accessing an additional table. The key for this additional table is the short code of the airline carrier.

The same applies for a range of further information. Airport information, technical data about aircraft and customer information is also stored in this way. This is how logical databases usually operate. The query provides utilities that allow you to access all this information.

Listed below are the most important tables, which contain additional information to that in the logical database F1S.

Table SCARR

This table contains additional information about an airline carrier, particularly the full name (long text). The table key is the airline carrier short code, which is stored in all of the logical database tables.

Table SAIRPORT

This table contains additional information about airports, particularly a long text. The table key is the airport short code. These short codes for airports occur in the table SPFLI twice for each record (for departure airport and destination airport).

Table SAPLANE

This table contains the technical data of a plane such as number of seats, fuel capacity, fuel consumption, dimensions, manufacturer and the like. The table key is the plane type short code. This short code occurs in the table SFLIGHT for each record. It indicates which aircraft is used on particular flight.

Table SCUSTOM

This table contains data about the customer such as name, address, telephone number, and other information. The table key is the customer number. Each record in SBOOK contains the number of the customer who made the booking.

QuickViewer

Purpose

The QuickViewer allows you to define reports without having to program yourself. The QuickViewer is especially useful for new users and occasional use.

Integration

QuickViewer is a tool for generating reports. SAP Query offers the user a whole range of options for defining reports. SAP Query also supports different kinds of reports such as basic lists, statistics, and ranked lists. QuickViewer, on the other hand, is a tool that allows even relatively inexperienced users to create basic lists.

QuickView definitions are user-dependent. You can transfer a QuickView into SAP Query in order to make reports, for example, accessible to additional users, or to use the other functions available in SAP Query.

The following is a comparison of QuickViews and queries:

- QuickViews possess the same functional attributes as queries. However, only basic lists may be defined with QuickViews.
- In contrast to queries, no user group assignment is necessary with QuickViews. Each user has his/her own personal list of QuickViews. QuickViews cannot be exchanged between users. QuickViews may, however, be converted to queries and then be made available to other users in a specific user group.
- InfoSets are not required for QuickView definition. Whenever you define a QuickView, you can specify its data source explicitly. Tables, database views, table joins, logical databases, and even InfoSets, can all serve as data sources for a QuickView. You can only use additional tables and additional fields if you use an InfoSet as a data source.
- The QuickViewer uses various controls. Certain hardware and software requirements must also be fulfilled before you can use the QuickViewer.

Features







To define a QuickView, you select certain fields according to your data source that determine the structure of your report. The report can be executed in basis mode with standard layout or may be edited using drag and drop and the other toolbox functions available in WYSIWYG mode.

Reports created using the QuickViewer may also be passed to external programs (Excel, for example).




Functions for Managing QuickViews

Call the QuickViewer using the menu path *System → Services → QuickViewer*.

Before you can execute one of the following functions, you must choose a QuickView, by either selecting it from the table control on your initial screen or entering its name in the appropriate input field.

Function:	Description:
 Display	Displays the QuickView definition details. All screens are displayed in the same manner as if the QuickView were being changed, however no new entries can be made.
 Description	You receive an overview containing general information about a QuickView and options. This includes: <ul style="list-style-type: none"> • QuickView title • Author and last person to make changes • Remarks specific to that QuickView • Origin of the selected data • Saved lists • Additional selections
 Copy	Choose a name for the QuickView in the dialog box and confirm your entry.
 Rename	Choose another name for the QuickView in the dialog box and confirm your entry.
 Delete	Deletes the QuickView after asking you to confirm that this is what you want to do.
<i>QuickView → More functions → Adjust</i>	<p>Adjusts differences between the QuickView and InfoSet.</p> <p>If you create QuickViews based on InfoSets, differences in individual field definitions may occur between a QuickView and its associated InfoSet.</p>  <p>Suppose you have created a QuickView using an InfoSet and have already executed it a number of times. Your system administrator then modifies an InfoSet field by changing the type of field you use in the QuickView. The next time you want to work with the QuickView, you get a message informing you that differences exist between the QuickView and the InfoSet.</p> <p>If differences occur between QuickViews and InfoSets, you should terminate processing and carry out a comparison between the QuickView and the InfoSet.</p> <p>When you carry out an adjustment you see a list of all the fields that are defined differently, together with information about how the field definition has changed and where the field is used in the QuickView. The system gives you an automatic adjustment option for individual fields. At the end of the list, there are some notes telling you what to look out for with automatic adjustment and what you must do, if you want to carry out the adjustment yourself.</p>

Functions for Managing QuickViews

 Layout display	The principle structure of the list created by the QuickView is displayed.
 SAP Query	<p>This pushbutton takes you to query maintenance.</p> <p></p> <p>To edit a QuickView in SAP Query, choose <i>Query</i> → <i>Convert QuickView</i> from the initial screen in SAP Query. For further information, see Converting QuickViews to Queries [Seite 360].</p>

Creating QuickViews

To convert a QuickView into a query, proceed as follows:

1. Call the QuickViewer using *System* → *Services* → *QuickViewer* (or transaction SQVI).
2. Enter the name of the QuickView. QuickView names can contain a maximum of 14 characters.
3. Choose *Create*. The following dialog box appears:

QuickView QUICKVIEWDEMO

Title Flight connections

Comments

1. Data source:

Table

2. Data from table/database view

Table/view

☒ Basis ... ☐ Layout mode

4. Enter a title for the QuickView and remarks, if you think they are relevant.
5. If you do not want to base your list on a table, use the possible entries pushbutton in the *Data source* field to select another data source. You can choose logical databases or InfoSets. In addition, you may also create table joins. For further information, see [Selecting a Data Source \[Seite 356\]](#).
6. Choose [Basis mode \[Seite 361\]](#) if you want to create the list directly with no list design. Choose [Layout mode \[Seite 367\]](#) if you want to define the layout of your list yourself.

Selecting a Data Source

Selecting a Data Source

The first step in creating a QuickView is selecting its data source. You can choose one of the following:

- Table
- Logical database
- InfoSet
- Table join

Logical databases, InfoSets, and table joins are all different techniques for consolidating data from multiple database tables. Logical databases are pre-defined paths for accessing database tables. These paths can be made available to various reports in code form. Logical databases are especially useful if the structure of the data you want to read corresponds to a hierarchical view. This can be realized using a logical database. More information on logical databases can be found under [Logical Databases \[Extern\]](#).

InfoSets are used in SAP Query. InfoSets can be based, among others, on logical databases. Since the number of fields in a logical database can be extremely large, you can also hide fields. In addition, you can define help fields that users can edit like database fields. You can attach additional tables to read long texts and store ABAP code, for example.

If you choose an InfoSet as your data source, the InfoSet must come from the standard query area. InfoSets from the global query area may not be used when creating a QuickView (consult the section on [Query Areas \[Seite 78\]](#) for more information). Also consult the section *Creating and Changing InfoSets* found in the SAP Query documentation if you are thinking about using an InfoSet.

Multiple tables can be linked together to form a join. The result set is a table, each of the lines of which contains all the fields of all the tables used in the join. Hierarchical relationships between tables cannot be analyzed using a table join. Table joins must be defined before beginning with QuickView construction. You define joins using a control containing graphical representations of the tables you are joining and of existing relationships between them. Further information is found under [Graphical Table Join Definition \[Seite 272\]](#).

Executing QuickViews

To execute a QuickView online, select *Execute* or *Execute with variant*. If you select *Execute with variant*, you see a dialog box where you can enter the variant name.



Variants are blocks of selection criteria that have been saved in the system. If you specify a variant when starting a QuickView, the system uses the values in the variant as the QuickView's selection criteria on the selection screen that appears.

You enter variants on the selection screen and save them under any name. For each QuickView, you can create any number of variants. To create a variant, select *Goto → Maintain variants*. You can find information about variant maintenance under *Help → Application help*.

The system starts the QuickView and displays the selection screen. Here you may enter selection criteria, just as if you were starting a report. Below is an example of a selection screen.

Selection screens normally consist of two parts:

- The upper part of the screen contains the database selections. These are determined by the underlying logical database and are automatically displayed on the screen.
- The lower part of the screen contains program parameters that come either from the QuickView's definition or that were automatically generated by the QuickView.

These program parameters may consist of any of the following:

- Additional selection criteria specified when defining the QuickView.
- Parameters and selection criteria defined, for example, in the InfoSet used.
- A parameter for the currency conversion date, if such conversions are performed in the QuickView.

Executing QuickViews

- Several radio buttons allowing you to send the list for further processing (see [Interactive List Display Functions \[Seite 109\]](#)).

When you have entered the selections, choose *Execute* to display your list.

If you want to output the list to a printer instead of on the screen, proceed as follows:

1. Start the QuickView as described above.
2. On the selection screen, choose *Execute and print*.

You then see a screen where you can specify the print options.

You can also display the list on the screen first and then select *Print*.

Output Options

Data retrieved by QuickViews and output in list form may be passed to other software products.



The QuickViewer's output options correspond to those of SAP Query.

In the Query Painter, you can determine how a list is output when you execute its report by selecting one of the appropriate radio buttons on the selection screen. You also select an output option when creating a query. Further information is found under [Assigning Title, Format, and Notes \[Seite 139\]](#).

In basis mode, you can use the possible entries help for field *Display as* prior to executing a QuickView to determine whether you want to export the report or display it directly.

For further information, refer to the following sections:

[Common Principles when Passing Lists for Further Processing \[Seite 118\]](#)

[Download to File \[Seite 119\]](#)

[Table Calculation \[Seite 120\]](#)

[Graphics \[Seite 121\]](#)

[Word Processing \[Seite 123\]](#)

[ABC Analysis \[Seite 124\]](#)

[EIS \[Seite 128\]](#)

[Private File \[Seite 129\]](#)

[Additional Function Pool \[Seite 130\]](#)

[Direct Interaction \[Seite 132\]](#)

Converting QuickViews to Queries

Use

You can convert any QuickView into a query. More functions and report forms are available in SAP Query than in the QuickViewer. In addition, SAP Query allows you to re-use transfer queries to other systems and clients.

Prerequisites

In order to convert a QuickView into a query, first choose a user group that you are assigned to and that you will subsequently want to assign your query to. If the QuickView you want to convert is based on an InfoSet, then this InfoSet must be assigned to your user group.

Procedure

To convert a QuickView to a query, proceed as follows:

1. Choose *Utilities* → *SAP Query* → *Queries* in the ABAP Workbench.
2. Choose *Change user group*. Select the user group you want.
3. Choose *Query* → *Convert QuickView*. The following dialog box appears:

The screenshot shows a dialog box titled "Convert QuickView". It has a light blue background. The dialog is divided into two main sections. The top section is labeled "of" and contains two input fields: "Quick View" with the text "QUICKVIEWDEMO" and "User" with the text "LANFERMANN". The bottom section is labeled "To" and contains three input fields: "Query" (a yellow field), "User group" (a grey field with the text "/SAPQUERY/FT"), and "InfoSet" (a white field). At the bottom of the dialog are two buttons: a green checkmark and a red X.

4. Enter a name for the query.
5. Enter a name of your choice for the InfoSet if the QuickView you want to convert has been created using a logical database, a table, or a table join. The InfoSet will then be generated automatically. Query functionality is based on InfoSets. In order to be able to proceed in SAP Query, an InfoSet must be defined for your query.

Basis Mode

Use

Use basis mode if you want to display or export your list in a standard format after field selection and sort sequence selection (if applicable).

Integration

You can choose between two modes in the QuickViewer:

1. Further processing in layout mode, or
2. List display in basis mode

Layout mode allows you to influence the graphical structure of your list, whereas basis mode immediately executes or passes your list to other tools.

Features

Basis mode offers you the following features:

- Field selection and sequence assignment
- Sort field selection
- Selection criteria selection (fields for the selection screen)
- List execution List display in standard format or list export (to Word, Excel, etc.)

Activities

If you want to create a QuickView in basis mode, follow the steps described in the section on [Creating QuickViews \[Seite 355\]](#).

For further information about specific procedures, refer to the following sections:

[Selecting List Fields \[Seite 362\]](#)

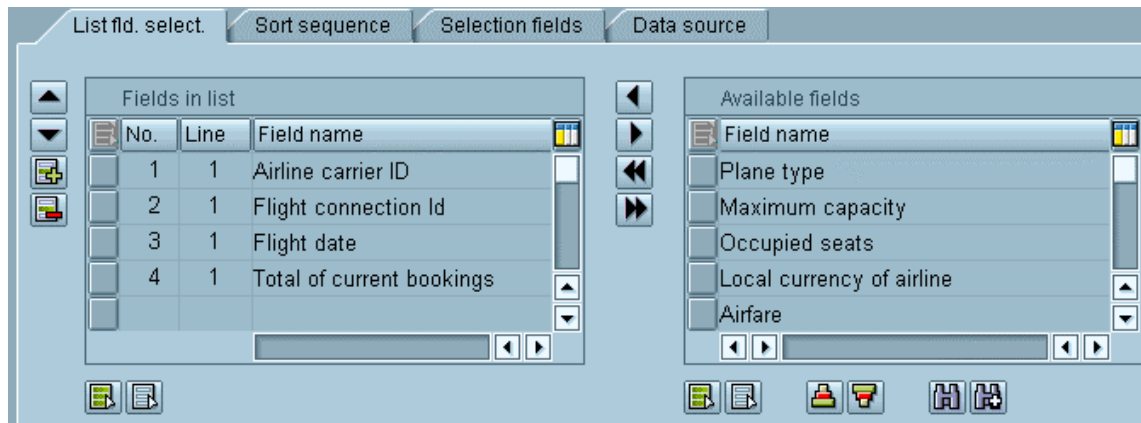
[Determining the Sort Sequencer \[Seite 363\]](#)


[Selecting Selection Fields \[Seite 364\]](#)

Selecting List Fields

Selecting List Fields

You choose what fields you want to appear on your list on the following screen.



Select *Available fields* that you need for your list from the table control on the right. The pushbutton  (*Column left*) allows you to transfer fields into the table control on the left.

If you want to accept all available fields in your list, use the *Page left* pushbutton (double-arrow). *Page right* allows you to reset this selection.

Determine in which order you want your fields to be output. If you want to move a field forward in the sequence, select it and choose the *Previous value* pushbutton (Arrow pointing upwards). Use the downward arrow to pass fields down the list.

Insert line allows you to determine where a line should wrap in multiple-line basic lists.




The *Technical name <> Long text* pushbutton in the application toolbar allows you to switch between a field's technical name and long text whenever you want to.

Determining the Sort Sequence

Choose the tab *Sort sequence*. Here you can determine list sorting and output criteria.

No	Asc/Desc	Fld name
1	<input checked="" type="radio"/>	Total of current bookings

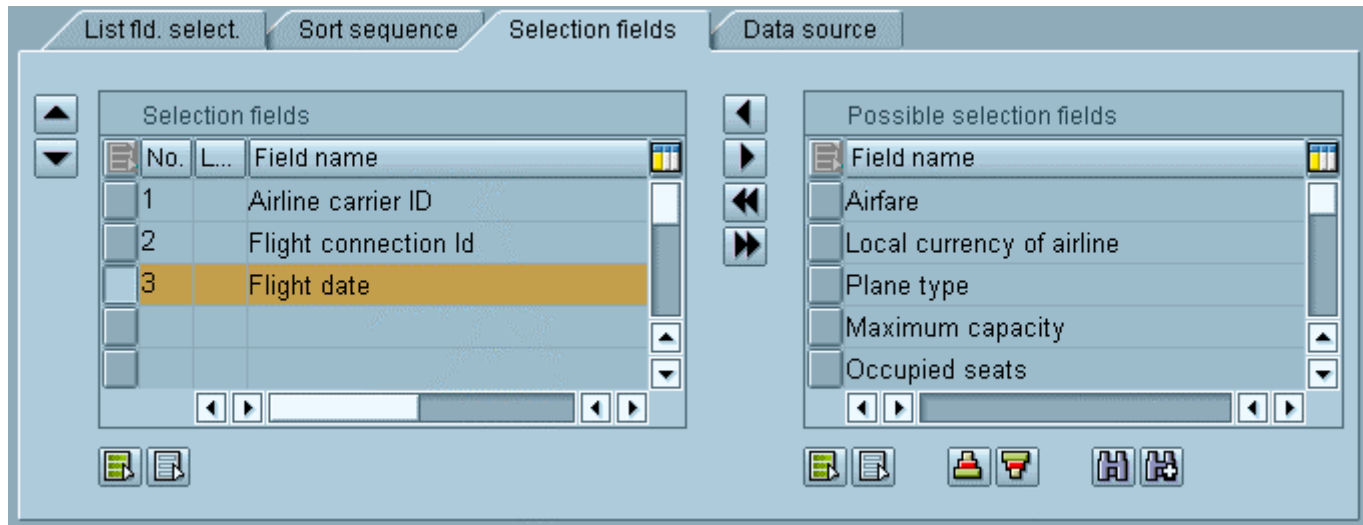
Fld name
Airline carrier ID
Flight connection Id
Flight date
Airfare
Local currency of airline

Select sort fields in the table control located on the right. You can sort and find fields in the *Available fields* table control with the appropriate pushbuttons. When you have found the fields you want to sort according to, use the  pushbutton (*Column left*) to transfer these fields to the table control on the left. Radio buttons allow you to determine if you want to sort ascending or descending.

Selecting Selection Fields

Selecting Selection Fields

Use the tab *Selection fields* to choose selection criteria in basis mode. Selection fields are displayed as input fields on selection screens before your report is executed. Users can reduce the amount of data output in the report by entering values in these fields. If, for example, you select *Flight connection code*, you can specify in the program selection area that you only want to display flights with certain flight numbers.





If you want to add a selection for a field, select that field in the right table control and transfer it to the table control on the left using the arrow key. Selection criteria for InfoSets or logical databases are already inserted in the left table control for you.

Example

You want to create a list in basis mode that displays information about airline ticket sales for specific airlines. This list should evaluate the data at hand and be displayed in the ABAP List Viewer. Proceed as follows:

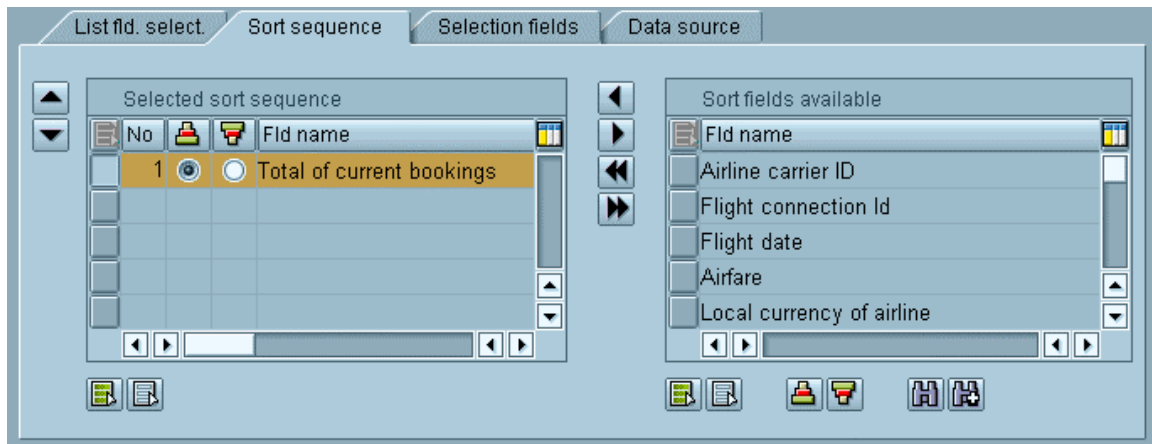
1. First, create a QuickView with logical database F1S as its data source. Further information can be found under [Creating QuickViews \[Seite 355\]](#).
2. Choose the following fields for your QuickView:

			Line 1
	1	1	Airline carrier ID
	2	1	Flight connection Id
	3	1	From city
	4	1	Destination
			Line 2
	5	2	Occupied seats
	6	2	Total of current bookings



Lines wrap automatically and are only relevant in layout mode.

3. Sort your list according to the fields *Flight connection ID* and *Total of current bookings*. To do so, choose the tab *Sort sequence* and transfer these two fields into the table control on the left.



4. Enter *ABAP List Viewer* in the *Display as* field located above the tab by using the possible entries pushbutton.
5. Choose *Execute*.
6. Enter the name of the airline carrier you want to use in the *Airline carrier* field on your selection screen.

A list similar to the following one appears:

Example

Tickets Sold						
ID	No.	From city	Dest.	Occupied	Booking total	Curr.
LH	0400	FRANKFURT	NEW YORK	0	0,00	DEM
LH	0400	FRANKFURT	NEW YORK	19	23.456,52	DEM
LH	0400	FRANKFURT	NEW YORK	66	81.878,04	DEM
LH	0400	FRANKFURT	NEW YORK	140	175.664,16	DEM
LH	0400	FRANKFURT	NEW YORK	233	291.268,44	DEM
LH	0402	FRANKFURT	NEW YORK	10	12.294,36	DEM
LH	0402	FRANKFURT	NEW YORK	16	20.379,60	DEM
LH	0402	FRANKFURT	NEW YORK	51	63.416,52	DEM
LH	0402	FRANKFURT	NEW YORK	62	77.868,72	DEM
LH	0402	FRANKFURT	NEW YORK	74	92.613,96	DEM
LH	2402	FRANKFURT	BERLIN	0	0,00	DEM
LH	2402	FRANKFURT	BERLIN	9	4.137,05	DEM
LH	2402	FRANKFURT	BERLIN	30	13.570,30	DEM
LH	2402	FRANKFURT	BERLIN	48	21.844,40	DEM
LH	2402	FRANKFURT	BERLIN	97	43.669,40	DEM

Layout Mode/The Graphical Query Painter

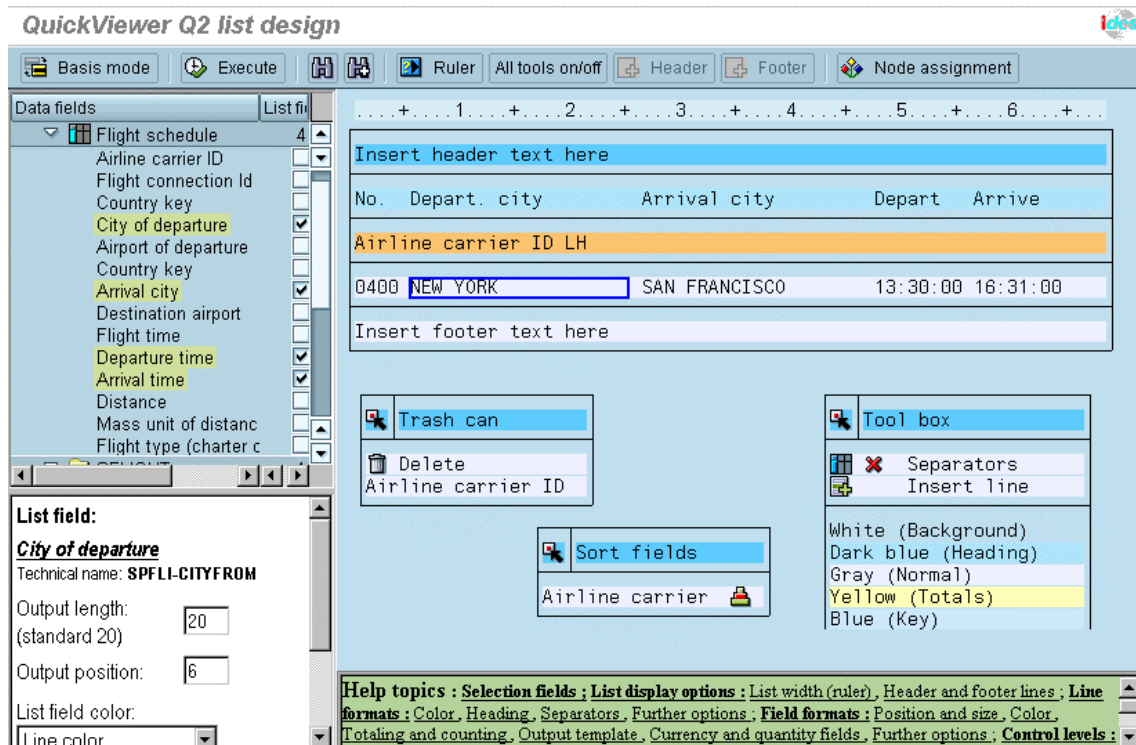
Use

The graphical Query Painter is called whenever you choose to define a QuickView in layout mode. It allows you to format a list according to your wishes and is used especially for constructing basic lists.

Features

In layout mode, you can:

- Define output options for lines
- Change the positioning and size of fields
- Create screen templates
- Edit headers, footers, and column headers
- Define control levels
- Sort
- Create totals lists etc.



The Query Painter is made up of screen divided into four windows whose size may vary according to wish. The individual windows contain

- Those fields at your disposal, organized on the left in tree form
- The list layout (on the right)
- Information about the list element currently selected (lower left)
- Links to the corresponding documentation as well as error messages and warnings (lower right)

Layout Mode/The Graphical Query Painter

The window on the upper left-hand side of your screen displays a list of all of the fields you can work with in this particular list. Further information can be found under [Selecting List Fields \[Seite 369\]](#).

List layout is simulated in the upper right-hand part of the screen. Individual fields are represented by field values. Sample data records are read from the data source. If this is not possible, dummy field values are inserted. This layout simulation corresponds to the structure of the list that will be created later. It contains the same fields, headers, colors, totals lines, etc. in the same order they will be displayed. In order to simulate the structure of a multiple-line, hierarchical list, numerous sample data records are read from the database and inserted in the list.

In addition, the layout window contains several [Toolbars \[Seite 397\]](#) for changing list layout and attributes.

Many of the elements that appear in the layout window can be moved and changed using drag and drop functionality. Your cursor changes from a pointer to an index finger when positioned on one of these elements. A simple left mouse click frames all moveable objects in blue and your cursor assumes a new shape (move cursor). Now you can move the element in question within the layout window by depressing your left mouse button and dragging the object to a new position. When you release the mouse button, your element assumes its new position (if the new position is allowed). Moveable elements include:

- All fields in the list layout
- Toolbars
- Fields appearing in toolbars
- Icons in toolbars

You may also change the width of certain selected elements (windows in the list layout, toolbars) by placing your cursor on the right edge of the element and increasing or decreasing its width with the left mouse button.

If an element is selected in the layout window, information about that element is displayed in the lower left window as soon as the blue frame appears around it in the layout window. With fields, long texts, technical names, and list attributes (display line, position, display length, sort number, etc.) are displayed. You may change these attributes directly in this window. Any entry you make in this window must be confirmed using the *Apply* pushbutton.

You may also call a context menu for an element using your right mouse button once that element has been selected in the layout window. This context menu allows you to display general list attributes or the attributes of specific list lines in the lower left window that cannot otherwise be simulated graphically. As with the other attributes, any entries or changes you make in this window must be confirmed using the *Apply* pushbutton.

For more information, see:

[Selecting List Fields \[Seite 369\]](#)

[Selecting Selection Fields \[Seite 370\]](#)

[List Display Options \[Seite 371\]](#)

[List Line Output Options \[Seite 375\]](#)

[Field Display Options \[Seite 380\]](#)

[Output Options for Control Levels \[Seite 390\]](#)

[Toolbars \[Seite 397\]](#)

Selecting List Fields

You can select list fields in the upper left window of the Query Painter. The fields are displayed in a tree whose structure is derived from that of the data source. If your data source is a logical database, then the nodes of this tree are the same as the logical database nodes. With joins, each table's individual nodes make up the node of the tree. All fields are found under their corresponding node.

Two radio buttons are assigned to each field. When you select the first radio button, its field is selected and transferred to the layout window, and as a result of this to the list as well. You can also make this selection simply by double-clicking on the field name. Selecting the second radio button makes the field into an additional selection criterion.

Selected fields are highlighted in color. Depending on how many fields you select, they are either inserted one after the other in a single line or in multiple lines.

If you do not want to fill every line, but instead want to move to the next line and resume filling there, you can insert new lines using the *Insert line* symbol in the toolbox. Those lines you have selected are then placed in the new line inserted.

To delete selected fields, undo the field's selection, or drag the trash can icon from the appropriate toolbar to the field you want to delete. Deleted fields are stored in the trash and can be re-inserted into the list according to need. The fields attributes remain the same as when they were deleted.



The system empties the trash whenever you exit the program.



Be aware that example data is used when structuring the list. The factual consistency of data in the layout mode is, in contrast to data displayed at list execution, not guaranteed.

Selecting Selection Fields

Selecting Selection Fields

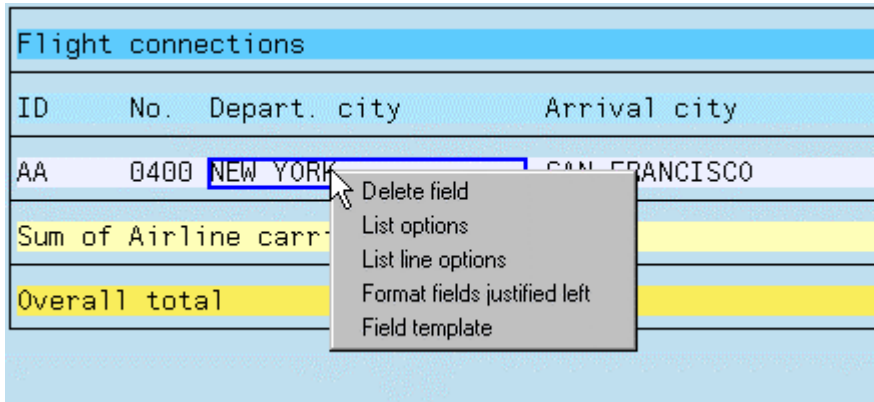
You can choose what selection fields you want to work with in the *Selection Fields* column of the upper left-hand window on your screen in the Query Painter. Select those fields that you want displayed as selection criteria on your selection screen by placing a check in their checkbox.

Selection fields are displayed as input fields on selection screens before your report is executed. Users can reduce the amount of data output in the report by entering values in these fields. If, for example, you select *Flight connection code*, you can specify in the program selection area that you only want to display flights with certain flight numbers.

Those selection fields mandated by your logical database or InfoSet are already selected.

List Display Options

Click on the ruler in the window on the upper right of your screen to determine list attributes. Several attributes may then be altered in the window on the lower left. You may also place your cursor on a list field and depress the right mouse button. The following context menu appears:



The screenshot shows a SAP Query window titled 'Flight connections'. It contains a table with columns: ID, No., Depart. city, and Arrival city. The first data row is highlighted in light blue and contains the values 'AA', '0400', 'NEW YORK', and 'SAN FRANCISCO'. A context menu is open over the 'NEW YORK' cell, listing the following options: 'Delete field', 'List options', 'List line options', 'Format fields justified left', and 'Field template'. Below the table, there are two summary rows: 'Sum of Airline carr-' and 'Overall total', both highlighted in yellow.

ID	No.	Depart. city	Arrival city
AA	0400	NEW YORK	SAN FRANCISCO
Sum of Airline carr-			
Overall total			

Choose *List options* and maintain your options in the lower left window. You can change list attributes here by making entries directly. Any entry you make in this window must be confirmed using the *Apply* pushbutton.

For more information, see:

[Headers and Footers \[Seite 372\]](#)

[List Width \(Ruler\) \[Seite 374\]](#)

Headers and Footers

Headers and Footers

Basic lists are displayed in the Query Painter header first, followed by the line structures, and subsequently a footer.

Two types of header lines are available: page headers and column headers. Headers and footers are independent of query structure. You can define any number of header lines. Blank header lines are not output in the list.

Column headers are made available to you by the system by selecting the checkbox *Display header* under the [List Line Output Options \[Seite 375\]](#). This checkbox is selected in the standard.

The following functions help you edit headers and footers:

- To insert text or parameters, click on the header or footer.
- To insert an additional line, double-click on the line. Each double-click inserts an additional line.
- To delete a line in your header, drag the trash can icon to that line.



If you have only a single header line and you delete it, you can re-insert a header by choosing *Additions* → *Insert header*.

You can define fixed header lines and footer lines so that when you generate the list, they receive current values. You can use the following fields directly as variables:

Function:	Abbreviation:	Description:
&%NAME	&N	Name of the user processing the query
&%DATE	&D	Current date
&%TIME	&T	Current time
&%PAGE		Current page number (6 characters)
&%P	&P	Current page number (3 characters)

Example

This is what a list formatted in the Query Painter looks like:

Flight connections				Date: &%DATE	
Depart. city	Arrival city	ID No.	Depart	Arrive	
NEW YORK	SAN FRANCISCO	AA 0017	13:30:00	16:31:00	
Time: &%TIME				&%PAGE	

At execution it looks like this:

Flight connections					Date: 06-01-1999	
Depart. city	Arrival city	ID	No.	Depart	Arrive	
NEW YORK	SAN FRANCISCO	AA	0017	13:30:00	16:31:00	
SAN FRANCISCO	NEW YORK	AA	0064	09:00:00	17:21:00	
NEW YORK	SAN FRANCISCO	DL	1699	17:15:00	20:37:00	
SAN FRANCISCO	NEW YORK	DL	1984	10:00:00	18:25:00	
FRANKFURT	NEW YORK	LH	0400	10:10:00	11:35:00	
FRANKFURT	NEW YORK	LH	0402	13:30:00	15:05:00	
FRANKFURT	SAN FRANCISCO	LH	0454	10:10:00	12:30:00	
SAN FRANCISCO	FRANKFURT	LH	0455	15:00:00	10:30:00	
FRANKFURT	BERLIN	LH	2402	10:30:00	11:35:00	
BERLIN	FRANKFURT	LH	2407	07:10:00	08:15:00	
BERLIN	FRANKFURT	LH	2415	09:25:00	10:30:00	
FRANKFURT	BERLIN	LH	2436	17:30:00	18:35:00	
FRANKFURT	BERLIN	LH	2462	06:30:00	07:35:00	
BERLIN	FRANKFURT	LH	2463	21:25:00	22:30:00	
ROME	FRANKFURT	LH	3577	07:05:00	09:05:00	
FRANKFURT	NEW YORK	SQ	0026	08:30:00	09:50:00	
NEW YORK	SAN FRANCISCO	UA	0007	14:45:00	17:55:00	
FRANKFURT	SAN FRANCISCO	UA	0941	14:30:00	21:06:00	
SAN FRANCISCO	FRANKFURT	UA	3504	15:00:00	10:30:00	
Time: 10:24:23					1	



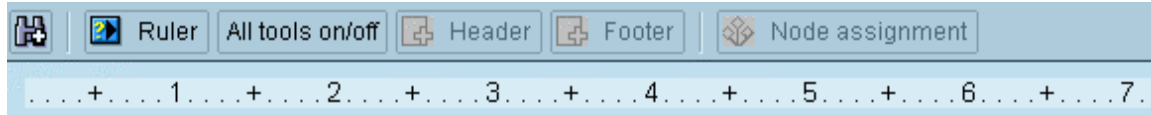
Please note that, although the number of header lines in the page header and footer is unrestricted, the total number of these lines must be smaller than the number of lines per page. You can only check this partially in the QuickView definition, since the number of lines per page may change at processing time (for example, when printing). If this condition is violated at processing time, the program is terminated.

List Width (Ruler)

List Width (Ruler)

The ruler is located on the edge of your screen at the upper right in the Query Painter. It is switched on in the standard.

You can hide or re-display the ruler on demand by using the appropriate pushbutton.



To set list width, proceed as follows:

- Place your cursor at the right end of the ruler. Once the frame appears, expand or contract it to the width you want. Alternatively, you may directly enter list width in characters in the *List width* field located in the lower left-hand part of your screen.



If you want to decrease list width so that it is smaller than the combined width of the fields on it, overhanging fields are inserted in a new line.

If you want to frame your list, select the checkbox *List with frame*.

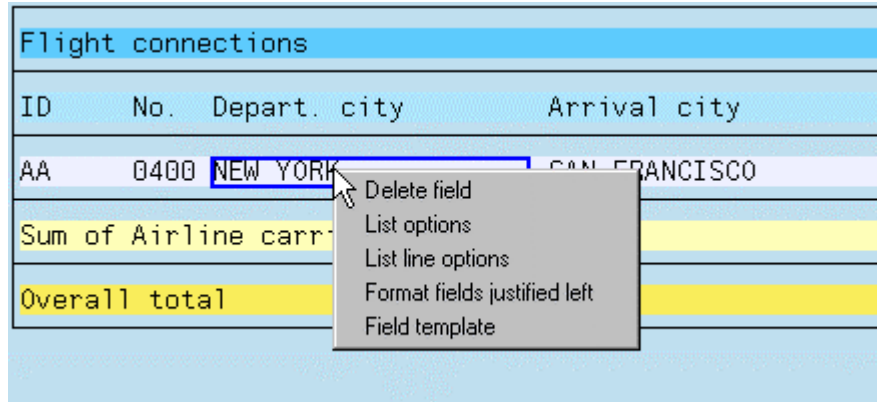


Your list must be framed if you want to use field separation.

List Line Output Options

There are two ways to define output options for a line:

- Place your cursor on the line in question and depress the right mouse button. The following context menu appears:



Choose *Line options* and maintain your line options in the lower left window.

or

- Double-click on an empty segment of the line (next to the list field). Those options that can be changed are now displayed in the lower left window on your screen.

For further information on line options, refer to the following sections:

[Color \[Seite 376\]](#)

[Column Headers \[Seite 377\]](#)

[Separators \[Seite 378\]](#)

[Further Options \[Seite 379\]](#)

Colors

Colors

You can use colors in lists to highlight specific information.

Select a color from the appropriate toolbox and drag it to the line you want (not to a list field). This line is now displayed in that color. In addition, you may also choose colors from the line options in the window in the lower left-hand corner using the possible values help. Selecting *Apply* then assigns the color to a line.

The Hierarchy Display

You may use the pushbutton *Node assignment* in the application toolbar of the Query Painter to highlight parts of your hierarchy display in color. When you choose this pushbutton, individual hierarchy levels are automatically displayed in different colors depending on their position in the logical database or functional area. This allows you to discern which fields belong to which level of the hierarchy and more easily order different lines these fields when creating multiple-line basic lists.

This highlighting is temporary and does not affect the colors you've selected for list display in any way. Choose *Node assignment* once again to switch off these colors. An example can be found in the section [Totaling and Counting Fields \[Seite 383\]](#).

SPFLI	11	No.	ID	Airp	From city	Apt	Dest.
Flight schedule	6	Flgt date	Plane	Capacity	Occupied	Booking	total
SFLIGHT	5	Airline AA					
Flight demo table	5	0017	AA	JFK	NEW YORK	SFO	SAN FRANCISCO
Airline carrier ID	<input type="checkbox"/>	11/19/1999	747-400	660	0	0,00	USD
Flight connection	<input type="checkbox"/>	11/22/1999	747-400	660	96	0,00	USD
Flight date	<input checked="" type="checkbox"/>	11/29/1999	747-400	660	48	0,00	USD
Ticket price	<input type="checkbox"/>	Total - Flight connection 0402					
Local currency of	<input type="checkbox"/>	0,00 USD *					
Plane type	<input checked="" type="checkbox"/>	Total - Airline LH					
Maximum capaci	<input checked="" type="checkbox"/>	0,00 USD **					
Occupied seats	<input checked="" type="checkbox"/>						
Total of current b	<input checked="" type="checkbox"/>						
SBOOK	0						

Many examples found in this documentation (the one above as well) are based upon logical database F1S. This database contains the tables SPFLI, SFLIGHT, and SBOOK. For each record (flight connection) in SPFLI, there are several records (flights for each flight connection) in SFLIGHT. For each record in SFLIGHT, the table SBOOK contains one record for each booking made.

See also the section [Logical Database F1S \[Seite 349\]](#).

Column Headers

You may display the structure of a basic list in the Query Painter: This display contains the header line (page header), the line structure, and the footer line (page footer).

Header lines are subdivided into page headers and column headers. The checkbox *Display header* is selected in the standard under [List Line Output Options \[Seite 375\]](#); this means that column headers will be displayed.

If you want to hide a line, deselect the checkbox for the entire line.

The system uses the short descriptions of fields stored in the ABAP Dictionary as column headers. If you want to change this text in a QuickView, do so by simply clicking on the column header you want to modify.



In queries, fields may only ever have one header.

Whenever you alter the output length of a field, the width of its header is changed as well.

Separators

Separators

Using separators makes list display clearer and more readily understandable.

No.	ID	Airp	From city	Apt
Flgt	date	Plane	Capacity	Occupied
Airline AA				
0017	AA	JFK	NEW YORK	SFO
11/19/1999	747-400	660	0	
11/22/1999	747-400	660	96	
11/29/1999	747-400	660	48	

Whenever you create a list in the Query Painter, it is framed automatically and its header and footer are set off from the body of the list with underscores.

You can decide if you want to display slashes before and after lines and if column separators make sense. These options are available as [List Line Output Options \[Seite 375\]](#).

You can also drag the separator icon from the toolbox to a specific line to insert column separators. Drag the *Undo* icon to that line to undo your choice.

Use this technique to undo the automatic selections mentioned above if you do not want your list's columns and lines to be separated. You can also delete the frame around your list on the [List Line Output Options \[Seite 371\]](#) screen. However, be aware that separators can only be used between individual column fields and lines if the list itself is framed.

Further Options

The following [List Line Output Options \[Seite 375\]](#) are available:

- *Line color*

You can assign a line a color in the following way:

- Call the possible values help for *Line color* from list line output options in the left lower window and choose *Apply*.
- Select a color from the toolbar for additional functions (toolbox) and drag it to the line you want.

The colors available are described by terms which identify their use in SAP standard displays.



You can also assign specific colors to the individual fields in a line independent of the line color.

- *Blank space before/after this line*

You can determine if you want a blank line to be inserted in front of or after a line. To do so proceed as follows:

- Enter the number of empty lines you want to have before or after the line under [List Line Output Options \[Seite 375\]](#) in the lower left-hand window on the screen. Confirm your entries using the *Apply* pushbutton.
- Select the *Insert line* icon from the toolbar for additional functions (toolbox) and drag it to the line you want. An empty line is inserted after each current line.

- *Redisplay line in header*

This option allows you to repeat certain lines in every page header.



For example, a query contains a flight connection (line 1) for multiple flights (line 2). This means that if you are working with a fixed page size (which is always the case when printing), one page may contain only lines with flights. However, if you select the option *Redisplay line in header* for line 1 (flight connection information), the line containing the flight connection will be output on each new page. Therefore, it is always easy to see to which flight connection the flights on each page belong.

- *New page*

If you select the option *New page* for a line, the system inserts a page break a new page before outputting this line.

- *Display only in connection with line n*

This option allows you to specify in connection with which line you want to output specific data. For example, flight connections (line 1) should only be displayed in connection with flights (line 2). In this case you must enter 2 in field *Display only in connection with line n* in the output options for line 1. You can now generate a list containing only those flight connections for which flights exist.

Field Display Options

A field's display options are displayed in the lower left-hand window whenever you select a particular field. These options include a long text and the object's technical name, as well as its attributes in the list (display line, display position, etc.). You can change a field's attributes directly by making entries in this window. Choose *Apply* to adopt new values. More information about these and other display possibilities can be found in the following sections:

[Positioning and Sizing \[Seite 381\]](#)

[Colors \[Seite 382\]](#)

[Totaling and Counting Fields \[Seite 383\]](#)

[Screen Templates \[Seite 387\]](#)

[Currency Fields and Quantity Fields \[Seite 388\]](#)

[Further Options \[Seite 389\]](#)

Positioning and Sizing

Moving fields

To change a field's position, proceed as follows:

1. Select the field.
2. Drag the field to the new position you want it at (keeping the left mouse button depressed).
Insert mode is the standard Query Painter setting, this means that whenever you position a field between to others, the system automatically makes room for the new field (without overwriting any existing fields). Overwrite mode can be switched on at will (Edit → Overwrite). Fields overlap in overwrite mode and a corresponding warning is displayed in the window at lower right. Use the *Order fields left-justified* command in the context menu (right mouse button) to display all overlapping fields so that you can see them.

If you want to move a field within a line, select the field and enter a number in the field *Output position* in the lower left window. The line will be displayed at this position. Use the ruler to help you position your fields.

Changing the Output Length of a Field

To change a field's output length, proceed as follows:

1. Select the field.
2. Place your cursor on the right edge of the field. Change the display length of the field by dragging the edge of the field to the length you desire.

You may also enter a value in the appropriate field in the lower left window on your screen to determine the output length of a field.



Increasing the output length may be particularly important if you are totaling a field and the output length required by the total is greater than that of the individual fields themselves.

Colors

Colors

Use the *Colors* toolbar to select a color and drag it to a field you want to highlight.

Or select the field using your mouse and choose a color from the *List field color* window in the lower left-hand corner of your screen. Possible entries help helps you select a color. Once you have chosen a color and applied it to a line, you can undo this selection by choosing *Line color* once again.



The colors available are described by terms which identify their use in SAP standard displays.

Totaling and Counting

Totaling

Whenever you create a list in the Query Painter, all numeric fields are automatically totaled. These totals are displayed at the end of the list. All numeric fields are also placed in the *Totaling fields* toolbar. To undo totaling for specific fields, select the fields from the toolbar and drag them to the trash or drag the trash to them or to the corresponding total in your list layout.

If sort fields have been defined for your list, subtotals are automatically displayed at the end of each control level. If you want to undo subtotalling for a specific sort level, select the trash can and drag it to the appropriate subtotal on the list layout display. To suppress all subtotalling functions, deselect the field *Display totals* in the [Output Options for Control Levels \[Seite 390\]](#).

If you are totaling a particular field, the total is output in the same column as the field, that is with the same output length. For this reason, the output length may be too short for the total and cause overflow. ABAP identifies these overflows by placing an asterisk in front of the affected value when it is output.

To avoid overflows like this when outputting totals, you can simply increase the output length of the field.

If you total fields containing values in different currencies, the totals are created according to currency.

Currency-specific summation also applies when you specify the output of the currency amount field without a unit. Then, several currencies (without a currency) appear in the totals lines. For this reason, you should always output currency amounts with a currency.

The same applies to quantity fields. If you specify summation for a quantity field, this is performed according to the units involved and results in a distribution in the totals lines.

Counting

In contrast to totals, counting fields are not automatically displayed. To count a field, proceed as follows:

1. Choose *Tools* → *Counting fields on/off*.

The *Counting fields* toolbar is not switched on in the standard.

2. Drag the field you want to count to the *Counting fields* toolbar.

The number of times this field appears is displayed at the end of the list. If sort fields have been defined for your list, preliminary counts are automatically displayed at the end of each control level. To suppress these preliminary counts, deselect the field *Display count* in the [Output Options for Control Levels \[Seite 390\]](#).

Finally, the options *Total* and *Counter* should be compared one more time.

The *Total* option causes the grand total of a specified numeric field to be calculated. This means that each time the field is read in a dataset, the field's value is added to the sum total. The grand total is displayed at the end of the basic list. With control levels, you can also display subtotals. A subtotal contains all values of a field that are assigned to a particular control level, that is a particular sort string.

The *Counter* option causes the counter number for a particular field to be increased by 1 every time that this field is found within the data set currently being read. The resulting counter total is displayed in much the same manner as a grand total at the end of the basic list. Just as with totals, counter subtotals can be displayed for individual control levels. These counter subtotals show how many values have been assigned to a particular control level.

Totaling and Counting

Example

You want to create a list that displays information about the airline ticket sales of a specific airline over a certain period of time. In addition, you want to display how many flights per flight connection are planned.

Proceed as follows:

- First, create a QuickView with logical database F1S as its data source. Further information can be found under [Creating QuickViews \[Seite 355\]](#).
- Select the following fields in the upper left-hand window in the Query Painter. *Airline ID*, *Flight code*, *Departure airport*, *Departure city*, *Arrival airport*, *Destination (city)*, *Flight date*, *Airplane type*, *Maximum occupancy*, *Occupied seats*, and *Current total of revenues*.
- Use the trash can to delete all sum totals except for the sum total for the field *Current total of revenues*.
- Drag the fields *Airline ID* and *Flight code* to the *Sort Fields* toolbar.
- Hide the control level text for *Flight code* by clicking on it and undoing its field selection in the [Output Options for Control Levels \[Seite 390\]](#).
- Drag the *Airline ID* field to the trash, since it is displayed in the control level text.
- Click on the control level text for *Airline ID* and change it to *Airline*.
- Switch on the *Counting fields* toolbar and drag the field *Flight date* to it.
- Choose *Node assignment*. The fields in the first line of the list have been taken from different levels of your logical database hierarchy.

SPFLI	11
Flight schedule	6
SFLIGHT	5
Flight demo table	5
Airline carrier ID	<input type="checkbox"/>
Flight connection	<input type="checkbox"/>
Flight date	<input checked="" type="checkbox"/>
Ticket price	<input type="checkbox"/>
Local currency of	<input type="checkbox"/>
Plane type	<input checked="" type="checkbox"/>
Maximum capaci	<input checked="" type="checkbox"/>
Occupied seats	<input checked="" type="checkbox"/>
Total of current b	<input checked="" type="checkbox"/>
SBOOK	0

No.	ID	Airp	From city	Apt	Dest.
Flgt date	Plane	Capacity	Occupied	Booking total	
Airline AA					
0017 AA	JFK	NEW YORK	SFO	SAN FRANCISCO	
11/19/1999	747-400	660	0	0,00	USD
11/22/1999	747-400	660	96	0,00	USD
11/29/1999	747-400	660	48	0,00	USD
Total - Flight connection 0402				0,00	USD *
Total - Airline LH				0,00	USD **

This means that the list is displayed as follows:

Totaling and Counting

No. Plane	Airp	From city Occupied	Apt	Dest.	Flgt date Booking total	Capacity
Airline LH						
0400	FRA	FRANKFURT	JFK	NEW YORK	11/29/1999	380
DC-10-10		19			23.456,52	DEM
0400	FRA	FRANKFURT	JFK	NEW YORK	12/02/1999	380
DC-10-10		140			175.664,16	DEM
0400	FRA	FRANKFURT	JFK	NEW YORK	12/09/1999	380
DC-10-10		66			81.878,04	DEM
0400	FRA	FRANKFURT	JFK	NEW YORK	12/29/1999	380
DC-10-10		0			0,00	DEM
0400	FRA	FRANKFURT	JFK	NEW YORK	12/31/1999	380
DC-10-10		233			291.268,44	DEM
Total - Flight connection 0400					572.267,16	DEM *
Number - Flight connection 0400					5	*
0402	FRA	FRANKFURT	JFK	NEW YORK	11/19/1999	380
DC-10-10		16			20.379,60	DEM
0402	FRA	FRANKFURT	JFK	NEW YORK	11/22/1999	380

The fields *Departure city, Destination (city) etc.* have to be displayed for each connection, even though appropriate positioning of the fields could make this unnecessary. Organize your fields in such a manner that only fields highlighted in red are found in the first line. Further information can be found under [Positioning and Sizing \[Seite 381\]](#) and in [The Hierarchy Display \[Seite 376\]](#).

- Change the texts for subtotals and preliminary counter totals by clicking on them.
- Insert separators (using the icon in the toolbox) and add additional slashes before and after list lines as you deem necessary. Further information can be found under [List Line Output Options \[Seite 375\]](#)
- Execute your QuickView. Enter an airline and a time period on the selection screen.
A list similar to the following might appear:

Totaling and Counting

No. Flgt date	Airp Capacity	From city Plane	Apt Occupied	Dest. Booking total	
Airline LH					
0400	FRA	FRANKFURT	JFK	NEW YORK	
11/29/1999	380	DC-10-10	19	23.456,52	DEM
12/02/1999	380	DC-10-10	140	175.664,16	DEM
12/09/1999	380	DC-10-10	66	81.878,04	DEM
12/29/1999	380	DC-10-10	0	0,00	DEM
12/31/1999	380	DC-10-10	233	291.268,44	DEM
Total - Flight ID 0400				572.267,16	DEM *
Number - Flight ID 0400					*
5					
0402	FRA	FRANKFURT	JFK	NEW YORK	
11/19/1999	380	DC-10-10	16	20.379,60	DEM
11/22/1999	380	DC-10-10	10	12.294,36	DEM
11/29/1999	380	DC-10-10	62	77.868,72	DEM
12/19/1999	380	DC-10-10	51	63.416,52	DEM
12/21/1999	380	DC-10-10	74	92.613,96	DEM
Total - Flight ID 0402				266.573,16	DEM *
Number - Flight ID 0402					*
5					
Total - Airline LH				838.840,32	DEM **
Total number - Airline LH					**
10					

Screen Templates

1. Screen templates allow you to insert texts in your lists. To insert descriptive texts in your lists, proceed as follows: Select the field you want to create a screen template for in the window on the upper right.
2. Increase the field's display length in lower left-hand window so that the number of characters you want to insert fits into the field.



Most of the time it will make sense reserve the field's output length for those values that are to be replaced and to increase the field's output length by the number of characters needed to accommodate the descriptive text.

3. Depress your right mouse button.
4. Choose *Field template* from the context menu that appears.

The line is now ready for input and an underscore has been inserted below. The length of the underscore indicates the output length of the field, that is, the number of characters reserved for field value substitution.



In the example below, the airport codes have been inserted in parentheses. An output length of three characters has been provided for the airport code. These three characters are indicated by the underline (___) and may not be overwritten by additional entries.

5. Enter the text you desire.

Example

Several descriptive texts have been inserted in the following list:

Departures and Arrivals		
Flight - 0400		
Depart.: FRANKFURT	(FRA)	10:10:00
Arrival: NEW YORK	(JFK)	11:34:00
Flight - 0402		
Depart.: FRANKFURT	(FRA)	13:30:00
Arrival: NEW YORK	(JFK)	15:05:00
Flight - 2402		
Depart.: FRANKFURT	(FRA)	10:30:00
Arrival: BERLIN	(SXF)	11:35:00

Currency Fields and Quantity Fields

Currency amount fields and quantity fields are numeric fields which each have a currency or unit field assigned to them. The correct interpretation of the values in these fields depends on its currency or unit of measure. Therefore, you can decide whether you want their corresponding currency/unit of measure to be displayed before or after the field, or not at all. Proceed as follows:

1. Select the field.
2. Choose one of the options from the window on the lower left (*before*, *No currency field/unit*, or *after*).
3. Choose *Apply*.



The assignment of amount fields to unit fields is defined in the ABAP Dictionary and evaluated by the QuickViewer.

Further Options

- *Display field only if <> 0*

Select this option if you want a field to be left empty instead of displaying a zero.

Output Options for Control Levels

Output Options for Control Levels

If you select a specific sort sequence for a particular list field, the system automatically provides you with a text at the start of your control level and creates subtotals for the control level's numeric fields.

For example, you have selected the following fields.

The screenshot displays the SAP Query interface. On the left, the 'Data fields' pane shows a tree structure with 'SPFLI' and 'Flight schedule'. The 'List fields' pane shows a sequence of fields: '.....1.....2.....3.....4.....5.'. The 'Control level' pane is active, showing the control level 'Airline carrier ID'. The 'Sort direction' is set to 'in ascending order'. The 'Start control level' options are checked: 'Control level text', 'New line', 'New page', and 'Frame around control level'. The 'End control level' options are checked: 'Display totaling' and 'Display counting'. The 'Apply' and 'Reset' buttons are at the bottom. The main list area shows a table of flight connections with columns 'No.', 'ID', 'Depart. city', and 'Arrival city'. The first row is 'Airline AA'. The second row is 'Flight connection Id 0400'. The third row is '0400 AA NEW YORK SAN FRANCISCO'. The 'Time: &%TIME' label is at the bottom left. A 'Sort fields' toolbar is visible, showing 'Airline carrier' and 'Flight connectio' with icons.

This QuickView is sorted according to the fields *Airline ID* and *Flight connection code*. After you drag these two fields to the *Sort fields* toolbar, the control level texts automatically appear above your list fields. Subtotals are displayed below the list's numeric fields.

Whenever you choose a sort field in the toolbox, the display options of its corresponding control level are displayed in the window at the lower left-hand corner of your screen. Here you may also determine a sort direction for each sort field and select options for the beginning and end of the control level. For more information, see:

[Sorting \[Seite 391\]](#)

[Control Level Text \[Seite 392\]](#)

[Subtotals \[Seite 393\]](#)

[Counting Fields \[Seite 394\]](#)

[Further Options \[Seite 396\]](#)

Sorting

To sort, proceed as follows:

1. Select the field you want to sort according to in the upper right-hand window.
2. Drag this field to the *Sorting fields* toolbox.

If multiple fields have been dropped into the toolbox, their order determines in which order sorting will take place. The system sorts according to the first field, then according to the second, and so on.

If you want to change the order in which fields are sorted, you can do so by selecting and moving fields within the toolbox.

An icon appears after each field indicating the direction in which the field is sorted (ascending/descending). You can change the sort direction simply by clicking on the icon.

Sorting can be undone by selecting the field in the toolbox and dragging it to the trash.

Whenever a field is identified as a sort field, a control level text is displayed the field in the Query Painter. Subtotals are displayed for numeric fields at the end of each control level. If a counter has been attached to the field, a partial counter value (subtotal) is recorded at the end of each control level. Further information can be found under [Totaling and Counting Fields \[Seite 383\]](#) and in the section on [Control Level Texts \[Seite 392\]](#).


Control Level Text


Control Level Text

Once the [Sorting \[Seite 391\]](#) of a particular field has been determined, a control level text is created that appears above the fields in your list in layout mode.

This control level text is highlighted in purple ('lilac'). You cannot change this lovely color!

Insert header text here						
No.	ID	Depart.	city	Arrival city	Depart	Arrive
City of departure NEW YORK						
0400	AA	NEW YORK		SAN FRANCISCO	13:30:00	16:31:00
Insert footer text here						

 Sort fields

City of departur 

If you select a sort field, as in the example above, and want it to appear as the control level text, it makes sense to delete the field from the actual body of the list. If you do not want the sort field to be elevated to the control level text, delete the control level text using the trash can icon. To delete the control level text, click on the sort field in the *Sort fields* toolbox and deselect the option *Control level text* in the lower left window on your screen. This checkbox is selected in the standard.

Click on control level texts to edit them. The whole line (minus the sort field display length) can be filled with text.

Texts for subtotals and counting fields can be edited in the same manner.

Subtotals

Once the sorting of a particular field has been determined, a control level is created. At the end of this control level, subtotals appear for all numeric fields in the list. A subtotal contains all values of a field that are assigned to a particular control level, that is a particular sort string.

Delete those subtotals that you do not need using the trash can icon.

If you do not want subtotals to be displayed at the end of a control level, deselect the *Display subtotals* option under [Output Options for Control Levels \[Seite 390\]](#).

Example

The list displayed below is sorted according to the fields *Airline ID* and *Flight connection code*. The subtotals are created and displayed accordingly.

No. Flgt date	Airp Capacity	From city Plane	Apt Occupied	Dest. Booking total
Airline LH				
0400	FRA	FRANKFURT	JFK	NEW YORK
11/29/1999	380	DC-10-10	19	23.456,52 DEM
12/02/1999	380	DC-10-10	140	175.664,16 DEM
12/09/1999	380	DC-10-10	66	81.878,04 DEM
12/29/1999	380	DC-10-10	0	0,00 DEM
12/31/1999	380	DC-10-10	233	291.268,44 DEM
Total - Flight ID 0400				572.267,16 DEM *
0402	FRA	FRANKFURT	JFK	NEW YORK
11/19/1999	380	DC-10-10	16	20.379,60 DEM
11/22/1999	380	DC-10-10	10	12.294,36 DEM
11/29/1999	380	DC-10-10	62	77.868,72 DEM
12/19/1999	380	DC-10-10	51	63.416,52 DEM
12/21/1999	380	DC-10-10	74	92.613,96 DEM
Total - Flight ID 0402				266.573,16 DEM *
Total - Airline LH				838.840,32 DEM **

Counting Fields

Counting Fields

The *Counter* option causes the counter number for a particular field to be increased by 1 every time that this field is found within the data set currently being read. The resulting counter total is displayed in much the same manner as a grand total at the end of the basic list. Just as with sum totals, counter subtotals can be displayed for individual control levels. These counter subtotals show how many values have been assigned to a particular control level.

Unlike totaling, counting fields are not displayed automatically. Fields you want to display counting fields for must be dragged to the *Counter toolbar*. This toolbar can be displayed by choosing *Tools* → *Counting fields on/off*.



The *Counter* option can also be selected for non-numerical fields.

Example

In this example, suppose you want to generate a list which contains booking information for each flight. A subtotal of revenues per booking should be displayed for each flight and a line containing a sum total of everything should appear at the end of the list. In addition, you want to know how many bookings were made per flight.

1. Choose fields as shown below. Numeric fields are automatically totaled. Since you only want to total the field *Amount (in foreign currency)*, use the trash can to delete the totals for the other fields.
2. Sort the list according to the field *Flight date*. The flight date now appears at the beginning of each control level. Subtotals are displayed prior to each change of date.
3. In order to find out how many bookings have been made per flight, choose *Tools* → *Counting fields on/off*. The *Counting fields* toolbar appears. If the option *Control level end - Display counter* is selected in the [Output Options for Control Levels \[Seite 390\]](#) (this is the setting in the standard), the number of bookings is displayed for each control level and the total number of bookings is displayed at the end of the list.

Counting Fields

Bookings					01.06.1999
From city	Dest.	Air Nr	Depart.	Arrival	
Fl. date	Ticket price	Occupied	Free		
Booking	Customer name	C S	Amount (for.crrncy)		
00000041	Ruthenberg, K.	Y	2.370,36	DEM	
00000042	Martin	Y	1.485,56	USD	
00000043	SAP Italy SPA	C X	2.469,12	DEM	
Total Fl-Date 10.05.1999			67.456,38	DEM	*
			16.548.776	ITL	
			9.898,66	USD	
Number Fl-Date			43		*
00000001	Becker, L.	F	2.271,59	DEM	
00000002	Mueller, A.	Y X	2.395,05	DEM	
00000003	Ratlos, Rudi	Y	2.322.202	ITL	
00000004	Starr	F	2.419,74	DEM	
00000005	Jackson, M.	Y	1.485,56	USD	
00000006	SAP Danmark	C	2.098,75	DEM	
00000007	SAP Oesterreich	C	2.395,05	DEM	
00000008	Brucher, F.	Y	2.320,97	DEM	
Total Fl-Date 12.05.1999			13.901,15	DEM	*
			2.322.202	ITL	
			1.485,56	USD	
Number Fl-Date			8		*
Overall total			146.641,09	DEM	**
			35.395.301	ITL	
			22.632,02	USD	
Total number			94		**

Further Options

Further Options

Other options available in conjunction with control levels include:


- *New line*
An empty line is displayed before the control level.
- *New page*
A page break is inserted after every control level change, for example after a change of sort string.
- *Frame control level*
Individual control levels are framed.

Toolbars

The layout window contains several toolbars for changing list layout and attributes. A toolbar is made up of a frame containing a header and other elements depending on the type of toolbar it is (icons, for example).

Notes on the functions available from individual toolbars are displayed in the lower left window on your screen whenever you click on a toolbar's header. Consult the corresponding passages in the online documentation on the Query Painter for more detailed information.

The following is a list of the functions available for switching toolbars on/off and altering their position on the screen:

Hiding all toolbars	Choose the pushbutton <i>All tools on/off</i> to hide or redisplay all toolbars at the same time.
Hiding individual toolbars	<p>Right click on a toolbar's header after have selected the toolbar as a whole. A deactivation pushbutton appears for hiding that toolbar. Alternatively, you can use the entries found in the <i>Tools</i> menu to activate/deactivate individual toolbars.</p> <p></p> <p>The toolbars <i>Toolbox</i> and <i>Trash can</i> can be displayed horizontally. Right click their headers and then choose <i>Horizontal</i>.</p>
Moving toolbars	Clicking on a toolbar's header changes your pointer into a move cursor (cross). You can now reposition this toolbar.



The *Counting fields* toolbar is not switched on in the standard. To do so, choose *Tools* → *Counting fields on/off*.