

TAW10

ABAP Workbench Fundamentals part II

PARTICIPANT HANDBOOK INSTRUCTOR-LED TRAINING

Course Version: 16
Course Duration: 10 Day(s)
Material Number: 50136303



SAP Copyrights and Trademarks

© 2016 SAP SE or an SAP affiliate company. 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 SE or an SAP affiliate company.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. Please see <http://global12.sap.com/corporate-en/legal/copyright/index.epx> for additional trademark information and notices.

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

National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP SE or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP SE or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platform directions and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, which speak only as of their dates, and they should not be relied upon in making purchasing decisions.



Typographic Conventions

American English is the standard used in this handbook.

The following typographic conventions are also used.

This information is displayed in the instructor's presentation



Demonstration



Procedure



Warning or Caution



Hint



Related or Additional Information



Facilitated Discussion



User interface control

Example text

Window title

Example text



Contents

ix	Course Overview
1	Unit 1: Introduction to the ABAP Dictionary
2	Lesson: Describing the ABAP Dictionary
9	Unit 2: Data Types in the ABAP Dictionary
10	Lesson: Creating Domains and Data Elements
17	Exercise 1: Create Basic Data Types
23	Lesson: Creating Flat Structures
27	Exercise 2: Create Simple and Nested Structures
32	Lesson: Creating Table Types and Deep Structures
35	Exercise 3: Create Table Types
39	Exercise 4: Create Deep Structures
44	Lesson: Creating Type Groups
53	Unit 3: Database Tables
54	Lesson: Creating Transparent Tables
67	Exercise 5: Create Tables in the ABAP Dictionary
77	Lesson: Defining Cluster Tables and Pooled Tables
85	Unit 4: Performance During Table Access
86	Lesson: Creating Database Table Indexes
91	Exercise 6: Create Table Indexes
95	Lesson: Setting Up Table Buffering
109	Exercise 7: Set Up Table Buffering
119	Unit 5: Input Checks
120	Lesson: Creating Fixed Values
123	Exercise 8: Create Fixed Values
126	Lesson: Defining Foreign Keys to Perform Input Checks
135	Exercise 9: Perform Input Checks to Ensure Data Consistency
146	Lesson: Creating Text Tables
149	Exercise 10: Create a Text Table



159	Unit 6:	Dictionary Object Dependencies
160		Lesson: Differentiating Between Active and Inactive Dictionary Objects
165		Exercise 11: Differentiate Between Active and Inactive Dictionary Objects
169		Lesson: Identifying Dependencies with ABAP Dictionary Objects
173		Exercise 12: Identify Dependencies with ABAP Dictionary Objects
179	Unit 7:	Table Changes
180		Lesson: Performing a Table Conversion
189		Exercise 13: Perform a Table Conversion
194		Lesson: Enhancing Tables Using Append Structures
201		Exercise 14: Add Enhancements to a Table Using an Append Structure
209	Unit 8:	Views and Maintenance Views
210		Lesson: Creating Database Views
223		Exercise 15: Create a Database View
227		Lesson: Creating Maintenance Views
235		Exercise 16: Create a Maintenance View
240		Lesson: Creating View Clusters
243		Exercise 17: Create a View Cluster
251	Unit 9:	Search Helps
252		Lesson: Creating Search Helps
267		Exercise 18: Create Search Helps
271		Lesson: Applying Advanced Search Help Techniques
275		Exercise 19: Create Collective Search Helps
279		Exercise 20: Create Append Search Helps
289	Unit 10:	Selection Screens
290		Lesson: Implementing a Selection Screen
301		Exercise 21: Implement a Selection Screen
307		Lesson: Implementing Multiple Selection Screens
313		Exercise 22: Define Tabstrips on a Selection Screen
319		Lesson: Implementing Input Checks and Creating Variants
325		Exercise 23: Implement Input Checks and Create Variants

333	Unit 11:	Introduction to Screen Programming
334	Lesson: Explaining the User Dialog Programming Model	
343	Exercise 24: Create a Dialog Program	
347	Lesson: Introducing Screen Programming	
353	Lesson: Creating Screens and Screen Elements	
361	Exercise 25: Create a Screen	
371	Lesson: Modifying Screens at Runtime	
376	Lesson: Designing Screen Sequence	
381	Lesson: Calling a Dialog Box Dynamically	
383	Exercise 26: Call a Dialog Box	
393	Unit 12:	The Program Interface
394	Lesson: Explaining User Interfaces	
400	Lesson: Setting a GUI Title and a GUI Status	
409	Exercise 27: Create a GUI Status	
421	Unit 13:	Simple Screen Elements
422	Lesson: Creating Screen Elements for Output	
430	Lesson: Creating Input/Output Fields	
434	Lesson: Defining Checkboxes and Radio Button Groups	
439	Exercise 28: Create a Radio Button Group	
445	Lesson: Creating Pushbuttons	
455	Unit 14:	Screen Error Handling
456	Lesson: Handling Errors Using Dialog Messages and Field Input Checks	
467	Lesson: Handling Errors Using Navigation and Input Help	
471	Exercise 29: Check Input Values	
485	Unit 15:	Subscreens
486	Lesson: Creating Subscreens	
497	Exercise 30: Embed Subscreens	
511	Unit 16:	Tabstrip Controls
512	Lesson: Creating Tabstrip Controls	
518	Lesson: Programming Tabstrip Controls	
523	Exercise 31: Create Tabstrip Controls	



Course Overview

TARGET AUDIENCE

This course is intended for the following audiences:

- Application Consultant
- Data Consultant
- Development Consultant
- Industry / Business Analyst Consultant
- Technology Consultant
- Developer
- System Administrator





Lesson 1

Describing the ABAP Dictionary

2

UNIT OBJECTIVES

- Describe the functions of the ABAP Dictionary



Describing the ABAP Dictionary

LESSON OVERVIEW

This lesson provides an overview of the functional scope of the ABAP Dictionary.

Business Example

As a developer, you must explain the main capabilities of the ABAP Dictionary to a colleague. For this reason, you require the following knowledge:

- An overview of the functions of the ABAP Dictionary in the SAP system
- An understanding of the different ways of defining data objects and data types
- An understanding of the services provided by the ABAP Dictionary
- An understanding of the relationship between the ABAP Dictionary and the tools provided by the development and runtime environment



LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Describe the functions of the ABAP Dictionary

Functional Overview of the ABAP Dictionary

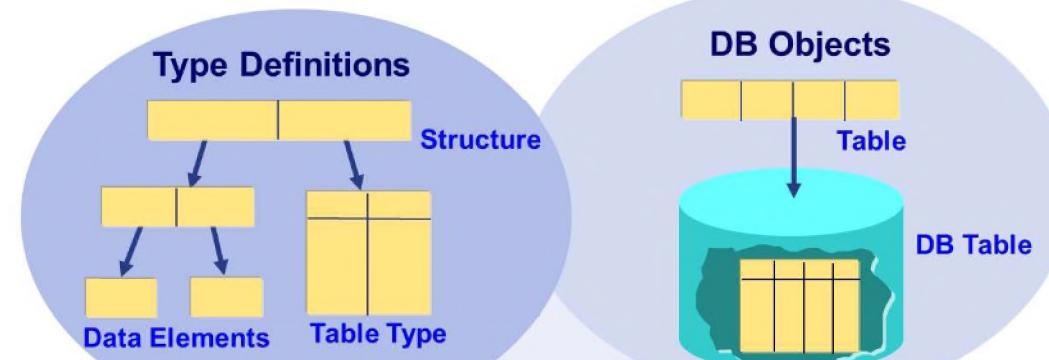


Figure 1: Function of the ABAP Dictionary

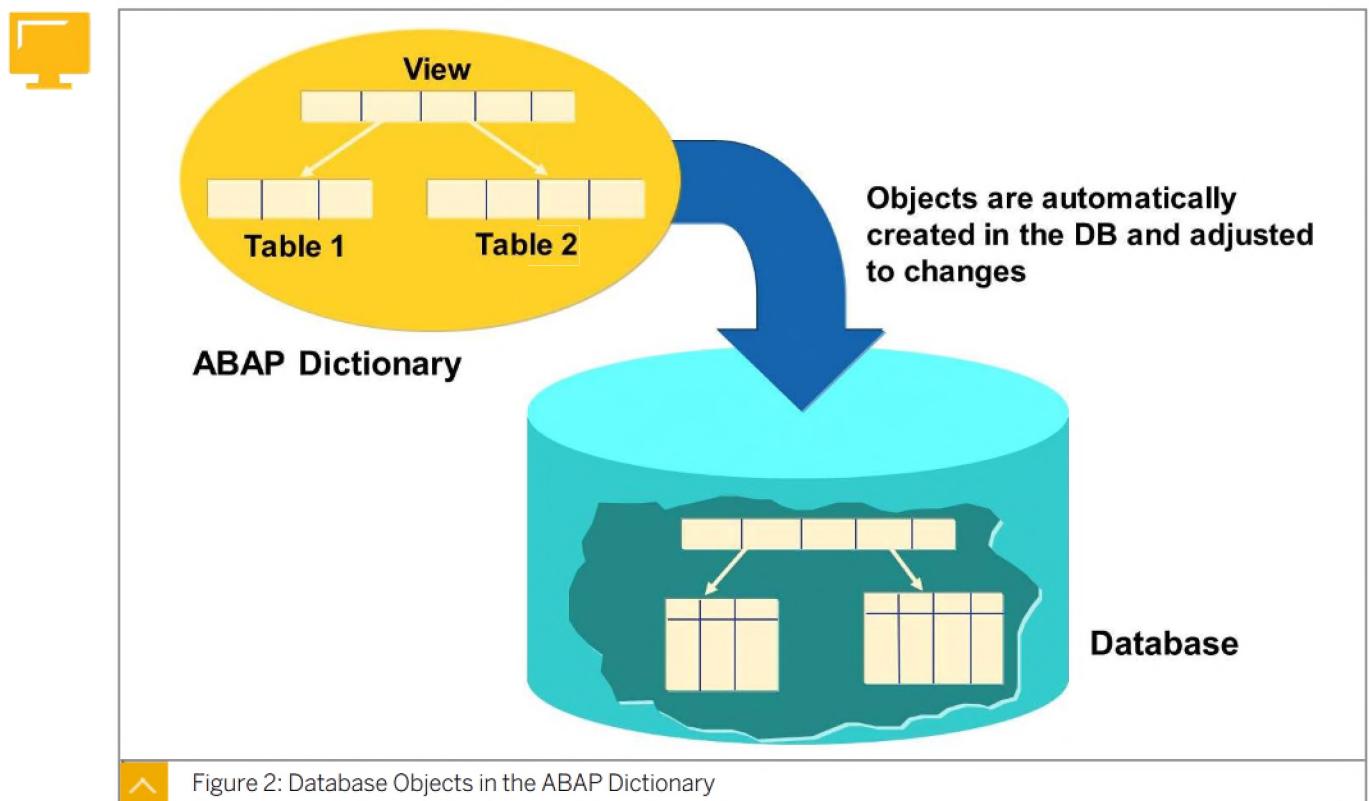


The ABAP Dictionary enables all the type definitions used in the SAP system to be managed centrally.

In the ABAP Dictionary, you can perform the following activities:

- Create user-defined types, such as data elements, structures, and table types, for use in ABAP programs or in interfaces of function modules, object methods, and so on.
- Create the database objects such as tables, indexes, and views in the ABAP Dictionary.
- Find a number of services that support program development. For example, the ABAP Dictionary supports setting and releasing locks, defining an input help (F4 help), and attaching a field help (F1 help) to a screen field.

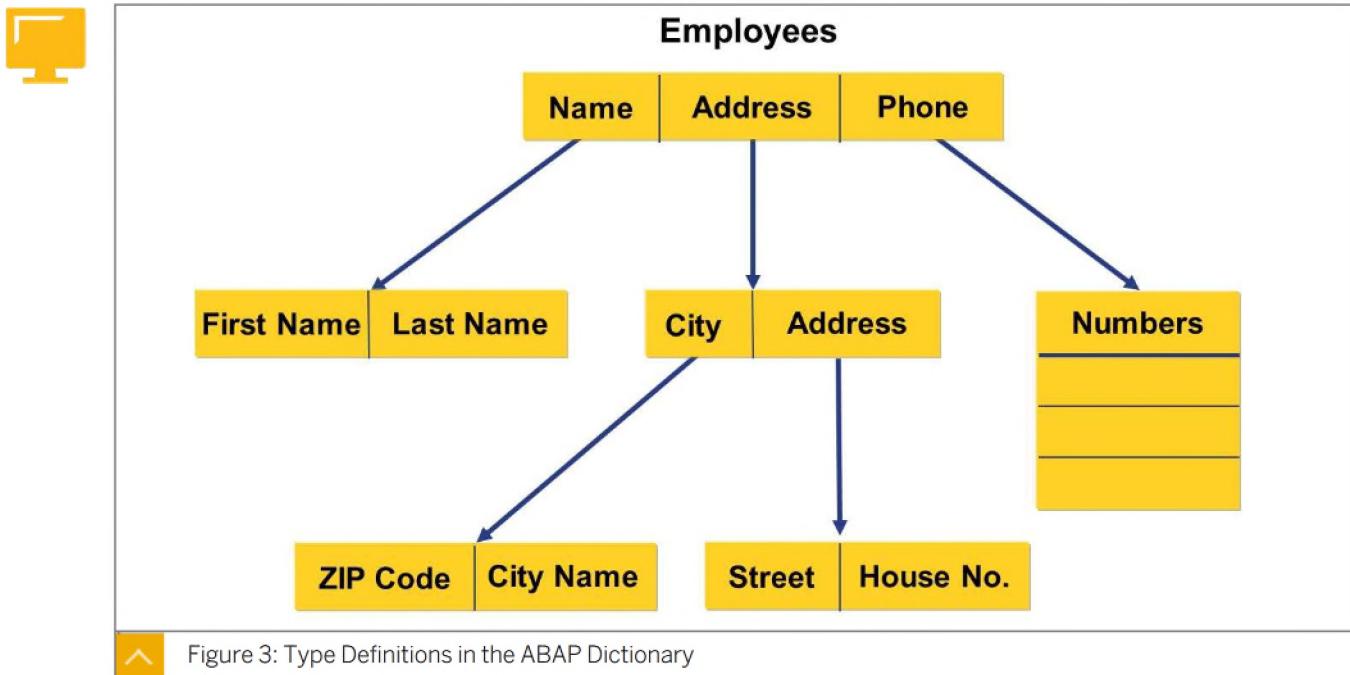
Database Objects in the ABAP Dictionary



In the ABAP Dictionary, you can perform the following functions:

- Define the tables and database views. The system creates these tables and views in the underlying database with the definition in ABAP Dictionary. Changes to the definition of a table or database view are also automatically made in the database.
- Define indexes in the ABAP Dictionary to speed up access to data in a table. The system also creates these indexes in the database when the table is activated.

Type Definitions in the ABAP Dictionary



Different type categories in the ABAP Dictionary are as follows:

- Data elements
Data elements describe an elementary type by defining the data type, length, and decimal places.
- Structures
Structures consist of components that can have any type.
- Table types
Table types describe the structure of an internal table.

Any complex user-defined type can be built from these basic types.

For example, the data for an employee is stored in a structure called Employee with Name, Address, and Telephone as its components. The component Name is also a structure, with First name and Last name as its components. The First name and Last name components are elementary, since the type is defined by a data element. The type of component Address is a structure, whose components are also structures. A table type is used to define the Telephone component because an employee can have more than one telephone number.

Types are used in ABAP programs at various places, for example, to define the types of interface parameters of function modules.

Services of the ABAP Dictionary

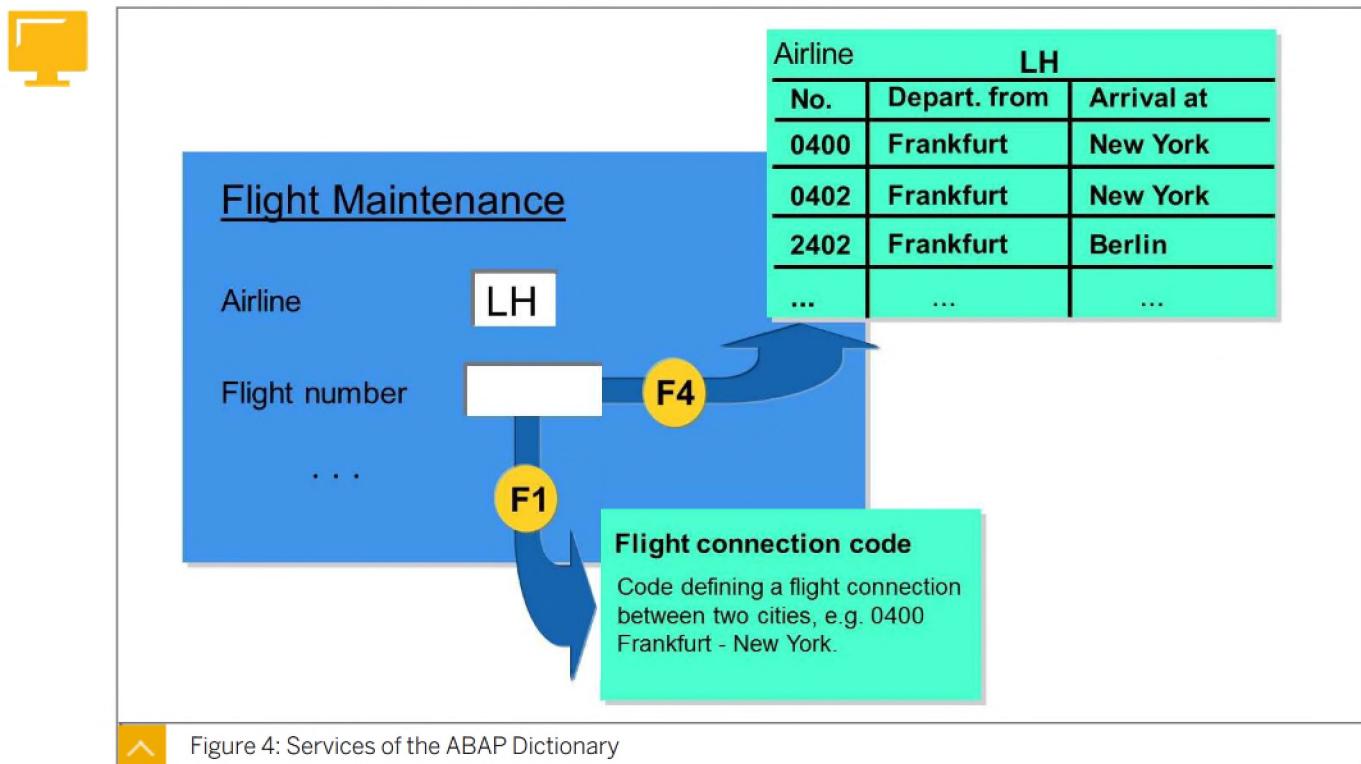
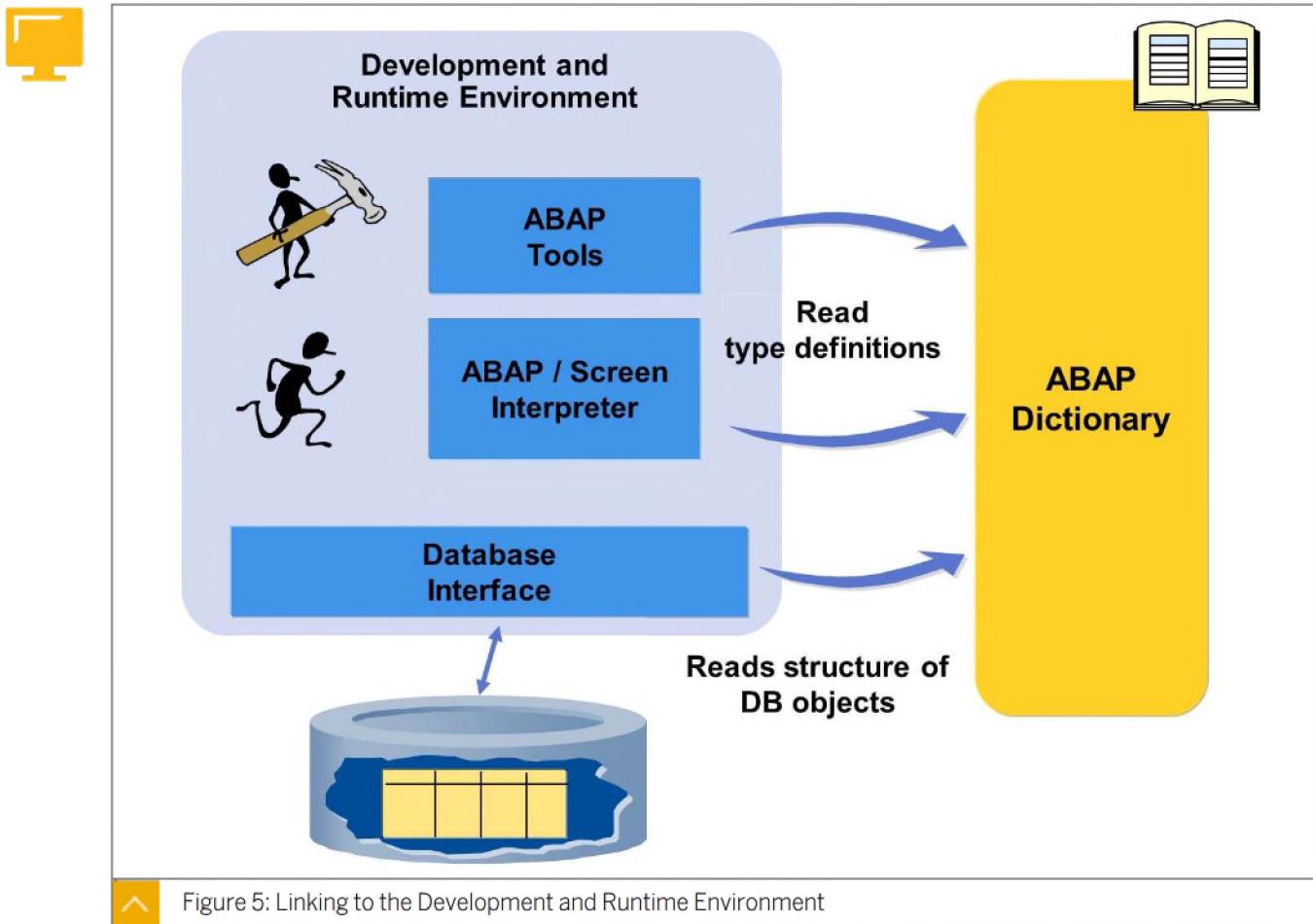


Figure 4: Services of the ABAP Dictionary

The following ABAP Dictionary services support program development:

- Input helps for screen fields
Input helps (F4 help) can be defined with search helps.
- Field help for screen fields
Screen fields can be assigned a field help (F1 help) by creating documentation for the data element.
- Input check
An input check ensures that the values entered are consistent and can be easily defined for screen fields using foreign keys.
- Set and release locks
The ABAP Dictionary provides support when you set and release locks. To set and release locks, you must create lock objects in the ABAP Dictionary. The function modules for setting and releasing locks are automatically generated from these lock objects; these can then be linked into the application program.
- Data buffering
Performance when accessing database data can be improved for database objects (tables, views, and so on) with data buffering.
- Logging
Logging allows you to record changes to the table entries automatically.

Linking to the Development and Runtime Environment



The ABAP Dictionary is actively integrated in the development and runtime environments. Each change in the ABAP Dictionary takes immediate effect in the relevant ABAP programs and screens.

The following examples explain the relationship between the ABAP Dictionary and the tools provided by the development and runtime environment:

- When a program or screen is executed, the ABAP Interpreter and the screen interpreter access the type definitions stored in the ABAP Dictionary.
- The ABAP tools and the Screen Painter use the information stored in the ABAP Dictionary to support you during program development. An example of this is the *Get from Dictionary* button in the Screen Painter, which you can use to place fields of a table or structure defined in the ABAP Dictionary on a screen.
- The database interface uses the information about tables or database views stored in the ABAP Dictionary to access the data of these objects.



LESSON SUMMARY

You should now be able to:

- Describe the functions of the ABAP Dictionary



Learning Assessment

1. Screen fields can be assigned a field help (F1 help) by creating documentation for the data element.

Determine whether this statement is true or false.

- True
- False

2. Which of the following type categories exist in the ABAP Dictionary?

Choose the correct answers.

- A Data elements
- B Structures
- C Indexes
- D Table types

3. When creating the user interface in the _____, fields from structures or tables defined in the Dictionary can be added to the screen using the *Get from Dictionary* function.

Choose the correct answer.

- A Function Modules
- B ABAP Interpreter
- C Screen Painter



Learning Assessment - Answers

1. Screen fields can be assigned a field help (F1 help) by creating documentation for the data element.

Determine whether this statement is true or false.

- True
 False

2. Which of the following type categories exist in the ABAP Dictionary?

Choose the correct answers.

- A Data elements
 B Structures
 C Indexes
 D Table types

3. When creating the user interface in the _____, fields from structures or tables defined in the Dictionary can be added to the screen using the *Get from Dictionary* function.

Choose the correct answer.

- A Function Modules
 B ABAP Interpreter
 C Screen Painter

UNIT 2

Data Types in the ABAP Dictionary

Lesson 1

Creating Domains and Data Elements	10
Exercise 1: Create Basic Data Types	17

Lesson 2

Creating Flat Structures	23
Exercise 2: Create Simple and Nested Structures	27

Lesson 3

Creating Table Types and Deep Structures	32
Exercise 3: Create Table Types	35
Exercise 4: Create Deep Structures	39

Lesson 4

Creating Type Groups	44
----------------------	----

UNIT OBJECTIVES

- Create domains for data elements
- Create data elements
- Create simple and nested structures in the ABAP Dictionary
- Create table types in the ABAP Dictionary
- Create deep structures in the ABAP Dictionary
- Define type groups in the ABAP Dictionary

Creating Domains and Data Elements

LESSON OVERVIEW

This lesson explains how to create domains and use them in data elements. It also explains data elements and how they are used as the basis for defining data objects in ABAP programs.

Business Example

You need to define simple and complex data types in the ABAP Dictionary, and to be able to use these data types in an ABAP program. For this reason, you require an understanding of the following:

- Data types
- Data elements
- Domains



LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Create domains for data elements
- Create data elements



Overview of Data Types

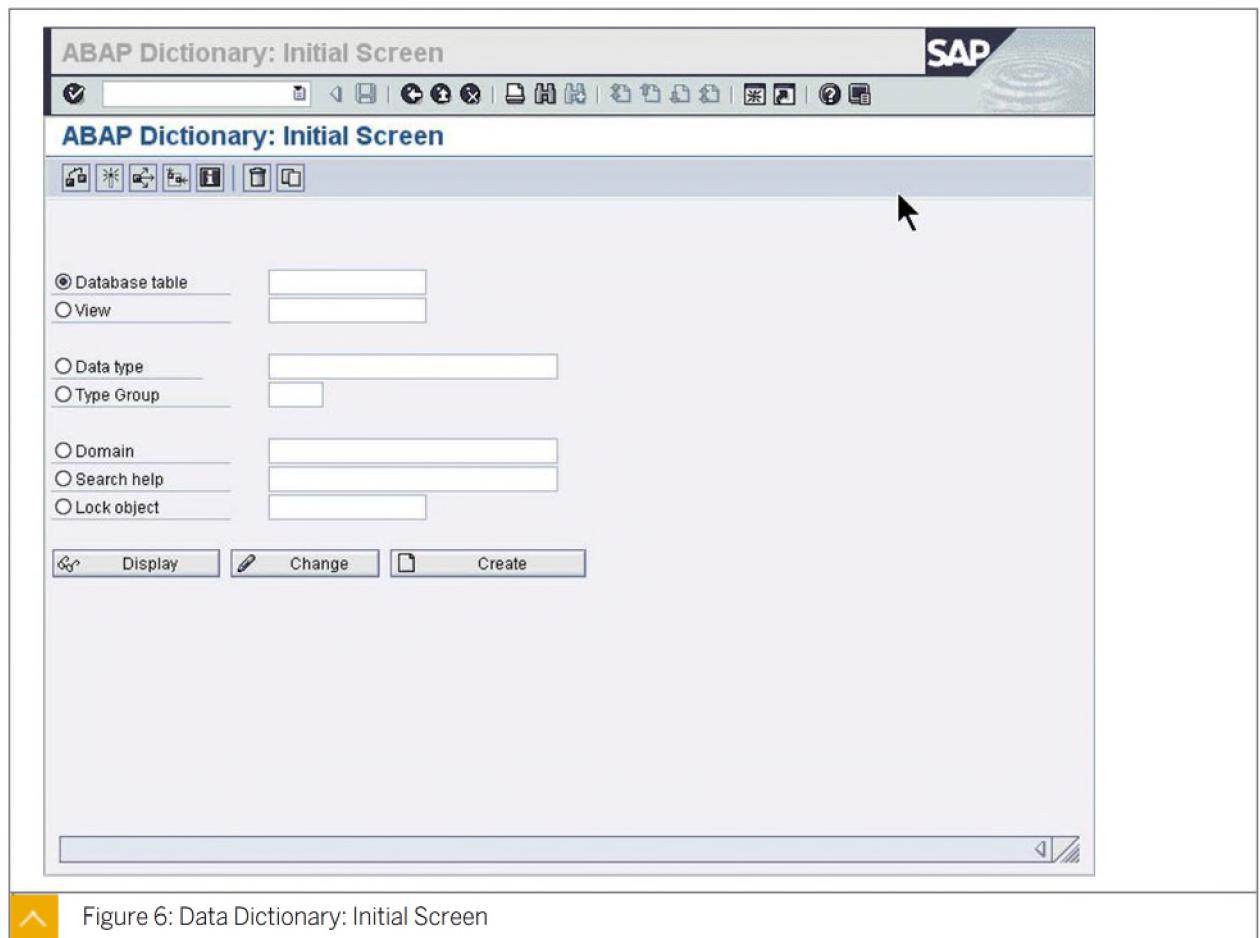


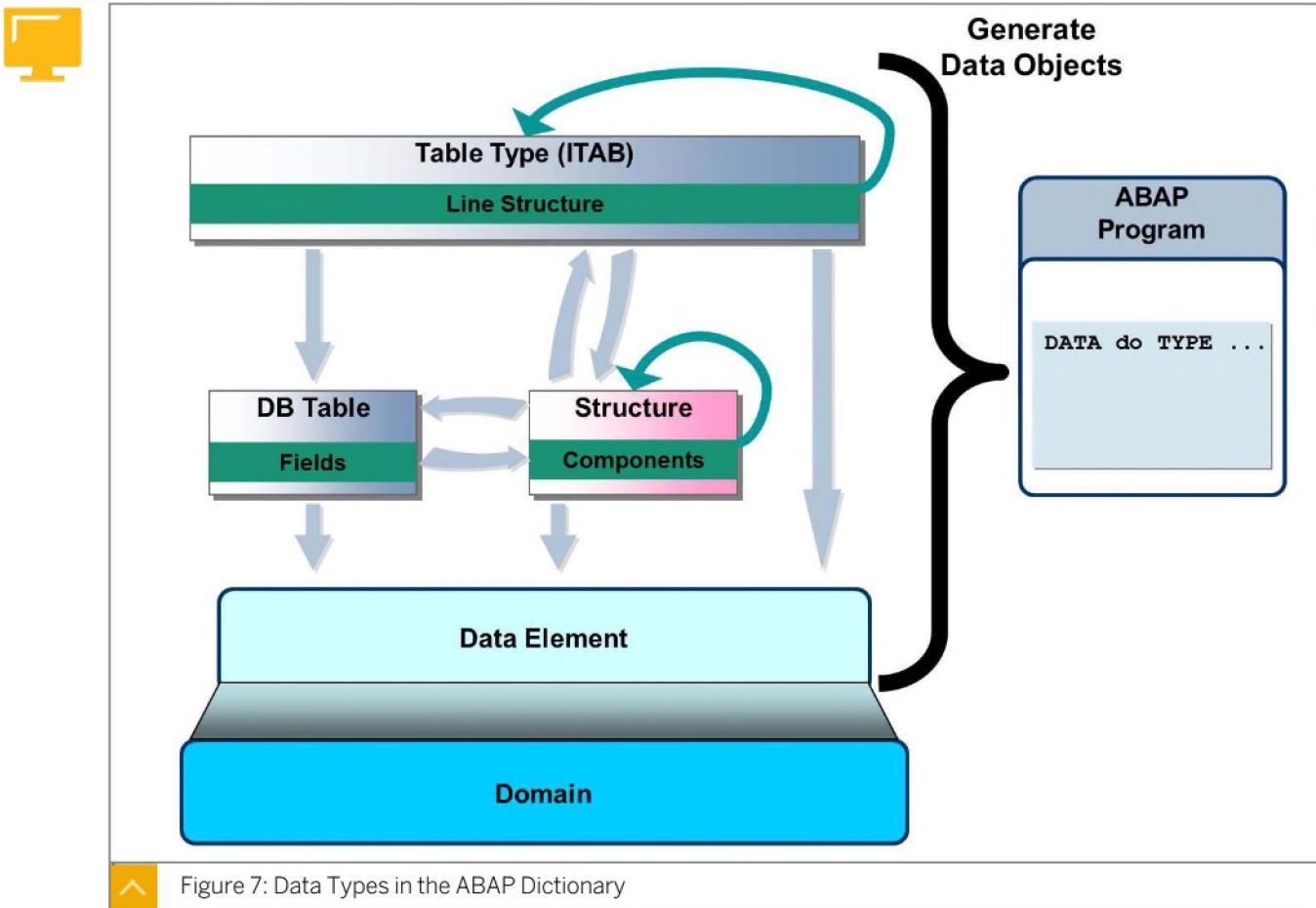
Figure 6: Data Dictionary: Initial Screen

Cross-program (globally known) data types can be defined in the ABAP Dictionary. You can refer to these data types in every ABAP program in the SAP system with the TYPE addition for the corresponding ABAP statements.

On the *ABAP Dictionary: Initial Screen*, in the *Data Type* field, you can find the following basic types:

- Data element
- Structure
- Table type

Data Types in the ABAP Dictionary



You can use the type definitions shown in the figure, Data Types in the ABAP Dictionary, (with the exception of the domain) to define data objects in ABAP programs. The arrows show how the type definitions can be used together.

Data elements use domains or integrated types to define their technical properties. However, an ABAP program cannot access domains to define data objects.

A structure consists of components that can be elementary fields, tables, and also structures.

The line type of a table type (ITAB) can be a structured type or an elementary type.

Domains

Domains manage the technical properties of data elements centrally. Domains cannot be used directly in programs, tables, and so on.

A Domain comprises the setting's format and output characteristics.

Format

In the format specifications, you find the data type and the number of characters that can be entered in the dependent data object. If the format is numeric, you can also specify the number of decimal places.

The data type must be chosen from a built-in list of data types.

The following are the most frequently used data types:

- CHAR (Character string)

Fields of this type can have a maximum length of 255 characters in tables. To use longer character strings in tables, you have to choose the `LCHR` data type. In structures, there are no length restrictions regarding these fields.

- **DATS (Date)**

The length for this data type is set to eight characters. You can define the output template with the user profile.

- **DEC (Decimal)**

A DEC field can have a maximum of 31 characters. This field is the calculation or amount field with decimal point, +/- sign, and thousands separators.

- **NUMC (Numeric)**

The length of NUMC field is restricted to a maximum of 255 characters. This character string can only contain numbers.

Output Characteristics

The maximum field length, including commas or decimal points, is specified for the input and output of values.

This value is usually calculated automatically once the *number of characters* has been assigned under *Format*. However, you can overwrite it later. The output format affects how screens and selection screens are displayed. The specifications from this area are used when a field is integrated in a screen. However, they can be modified in the Screen Painter.

You can also define a conversion routine. This conversion routine changes the display format (for example, place leading zeros before a number) when the content of a screen field is converted from the display format to the SAP-internal format, and vice versa. The same conversion routines are executed by the system when you use the ABAP statement `WRITE`. Similarly, you can also use this conversion routine to override any unsuitable standard conversions.

For certain data types, such as DEC, FLTP, QUAN, and CURR, the check box `+/ - sign` is ready for entry. If this is activated, the first character of the field is reserved for the `+/ - sign` on a screen. Accordingly, the system increases the output length by 1.

For character-based data types, you should also determine whether lowercase letters are allowed. If this flag is not set, you can enter lowercase letters in the respective input fields; however, the lowercase letters are transformed into uppercase letters as soon as the user confirms the entry (for example, with the `ENTER` key).

In addition, you can define valid value ranges that are used for input checks.

Data Elements

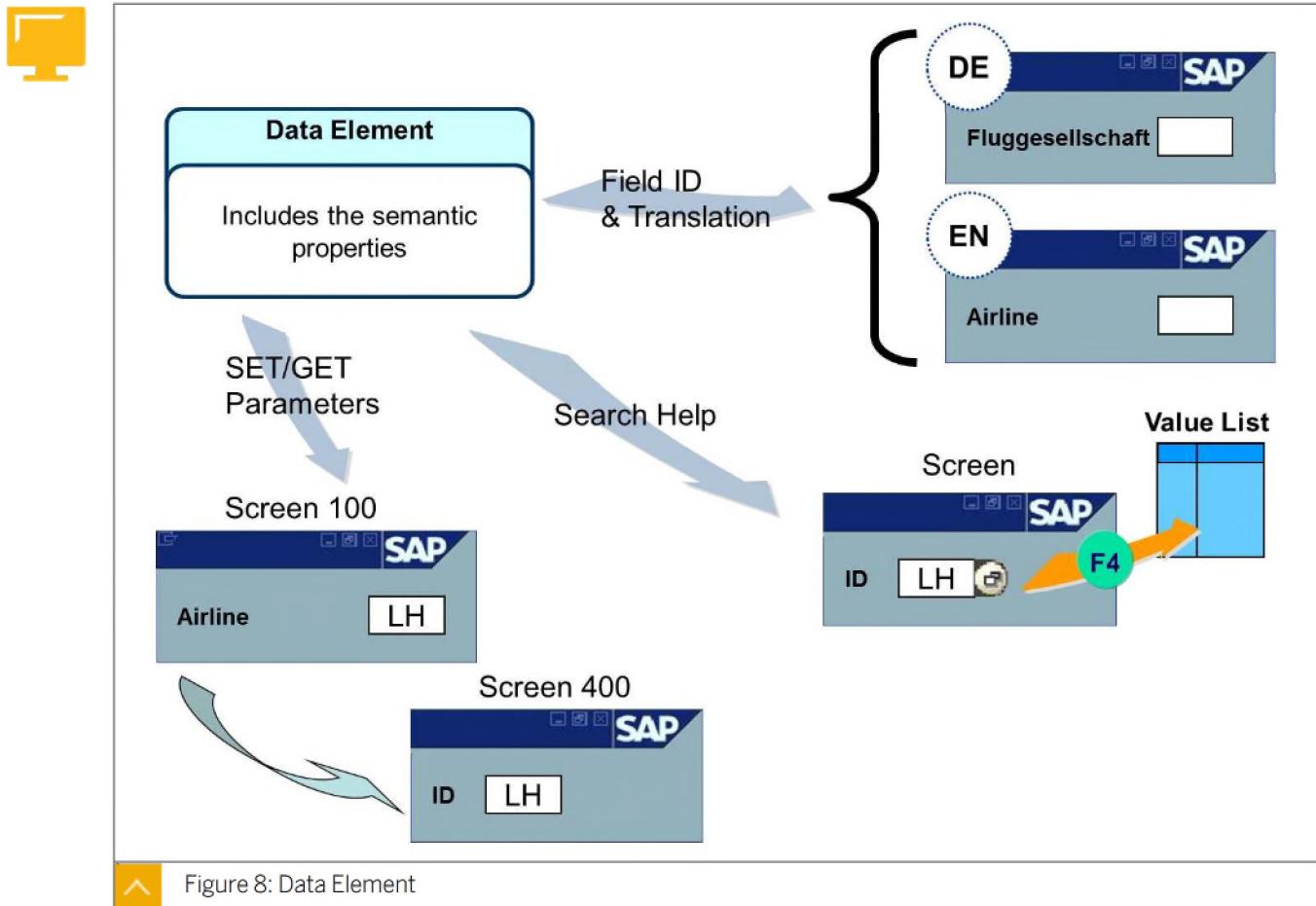


Figure 8: Data Element

Data elements define data types which can be used on screens, search helps, ABAP programs, and complex data types. Data elements contain both semantic and technical information about the data type.

The technical properties for the data element are maintained on the *Data Type* tab page. SAP recommends using domains to specify the technical type of a data element. However, you can also define the data element using the same integrated types that are used to define the domains. As a special case, you can also create a data element as a reference type. The referenced type may be any type defined in the dictionary or a generic reference to types ANY, OBJECT, and DATA. A reference of the type ANY can point to both objects and data. The definition as a reference is the same as the type declaration in an ABAP program `TYPES tr_dt TYPE REF TO data.`

The semantic information is maintained in the field labels. These field labels (short, medium, or long) can be displayed on screens or selection screens to explain the field content.

You also have to specify maximum length for the respective field label. If you work for a company that operates globally, you can translate the field labels into other languages. (On the *ABAP Dictionary: Initial Screen*, choose *Goto → Translation*, or run the transaction code SE63.) When specifying the maximum length, remember that the same term in the field label might require more letters in another language.

A search help (F4 key or input help) can be appended to a data element. The subject of search helps is discussed in greater detail later in this course. Search helps can be integrated at different levels.

Data Elements: Additional Options

Additional options for data elements are as follows:

- Documentation

You can create a text that describes the contents of the data element. This text is displayed for the F1 help in all screen fields that refer to this data element.

- Search help

A search help (F4 key or input help) can be assigned to a data element. The subject of search helps is discussed in greater detail later in this course. Search helps can be integrated at different levels.

- Set/Get Parameter

Assigning a SET/GET parameter to the data element saves the user from entering the same value several times. A field can be filled with default values from SAP memory. When you exit the screen, the system transfers the value into this parameter. If an input field based on the same data element exists on a subsequent screen, the system reads the value from the parameter and enters it in the screen field. The SET/GET parameters hold the value for each session. These values are not retained after the user has logged off.



Note:

The SET/GET parameters have nothing to do with the SAP GUI history (local data).

- Default component name

You can assign a default component name to the data element. However, this is effective only if you use the data element as a component in Business Application Programming Interface (BAPI) structures.

- Change document

The system logs changed field contents only if you set the Change document indicator for the data element.

- Input history

The mechanism that the SAP GUI uses to handle the input history of a screen input field is switched off.

- Bi-directional options

SET/GET Parameter

In different applications, you may need to enter a particular value in several screens.

Assigning a SET/GET parameter to the data element saves the user from entering the same value several times. When you exit the screen, the system transfers the value into this parameter. If an input field based on the same data element exists on a subsequent screen, the system reads the value from the parameter and enters it in the screen field. The SET/GET parameters hold the value for each session. These values are not retained after the user has logged off.

The SET/GET parameters have nothing to do with the SAP GUI history (local data). You can also assign a default name to the data element. However, this is only effective if you use the