Lection 2 Workbook

Data Dictionary objects
Part 1

Contents

Development objects and packages

When you work with the Workbench, you work with development objects and packages. Development objects are the individual parts of an ABAP application. Some examples of development objects are programs like reports, transactions, and function modules. Program components such as events, screens, menus, and function modules are also development objects.

The SAP system stores development objects in the Repository, which is a part of the database. When you complete work on a development object like a program, screen, or menu, you generate a runtime version of the object. This runtime version is stored, along with the object, in the Repository. An application consists of several runtime objects that are processed by the work processes in the SAP System.

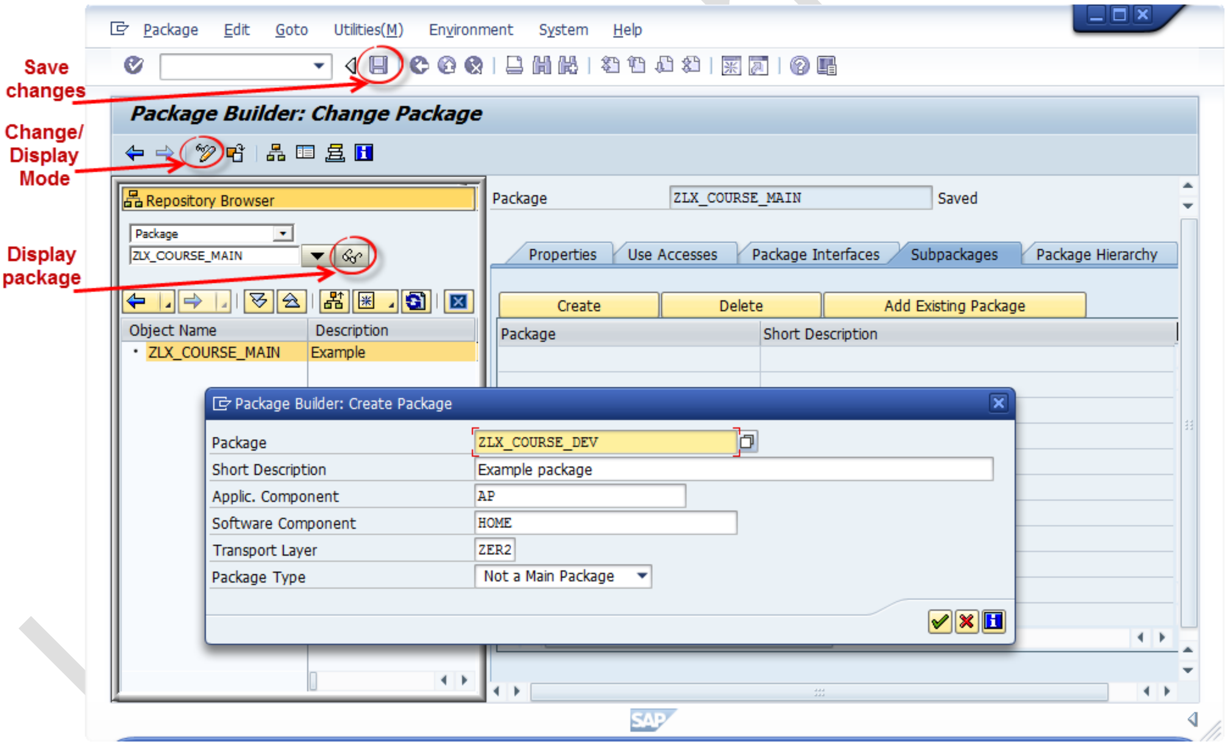
A package is a container for objects that logically belong together; for example, all of the objects in an application. A package is also a type of development object.

When you create a new object or change an existing object, the system asks you to assign the object to a package.

The main package is primarily a container for development objects that belong together, in that they share the same system, transport layer, and customer delivery status. However, you must store development objects in sub-packages, **not** in the main package itself.

You can add packages to a main package in one of two ways. If you wish, you can create new sub-packages – from within a main package – and assign them to the main package at the same time. Alternatively, you can add existing packages to a main package in a separate step.

How to create a package


Save changes

Change/ Display Mode

Display package

Package Builder: Change Package

Package: ZLX_COURSE_MAIN Saved

Object Name	Description
ZLX_COURSE_MAIN	Example

Package Builder: Create Package

Package	ZLX_COURSE_DEV
Short Description	Example package
Applic. Component	AP
Software Component	HOME
Transport Layer	ZER2
Package Type	Not a Main Package

To add new sub-package to the main package:

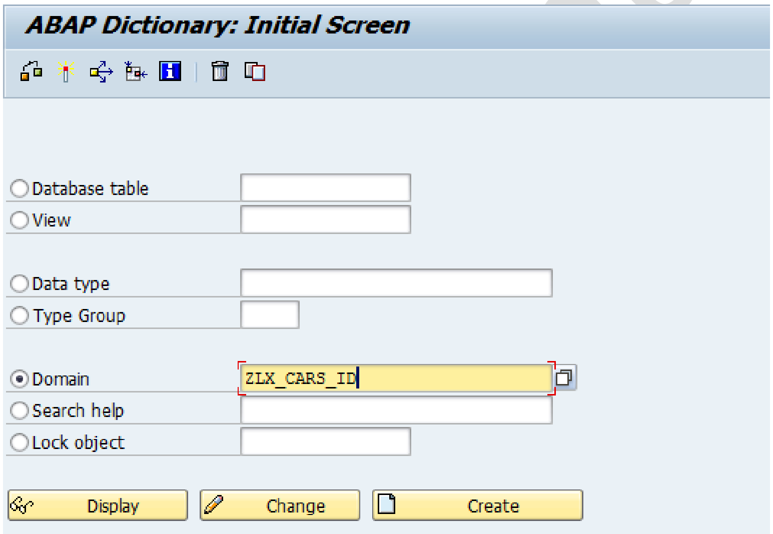
1. Go to Object Navigator in ABAP Development Workbench (transaction SE80).
2. The main package has already been created by your tutor. Find the relevant main package in the Repository Browser:
    a. Choose the Category of Development Object – Package.
    b. Type in the name of the relevant main package and choose Display.
3. By double-clicking on the displayed package you will open it in the Package Builder.
4. Make sure you switch to the Change mode.
5. Choose the Subpackages tab.
6. To create new sub-package, choose the Create button. The system displays the Create Package dialog box.
7. Enter attributes (from the table below) for this package and choose Execute.

Package:	**Z***_DEV** (*** is your namespace)
Short Description:	**Package of ***user name*****
Application Component:	**AP**
Software Component:	**HOME**
Transport Layer:	**ZER2**
Package Type:	**Not a Main Package**

8. Save your package.

ABAP Dictionary

Data Dictionary (DDIC) is a tool of ABAP workbench, is used to store the description of data definition. It works as an interface between user and SAP system to create Tables, Views, Structures, Data elements, Domains, Search helps, Lock Objects etc. Data definitions (metadata) are created and managed in the ABAP Dictionary (transaction SE11).



ABAP Dictionary: Initial Screen

○ Database table
○ View
○ Data type
○ Type Group
● Domain ZLX_CARS_ID
○ Search help
○ Lock object

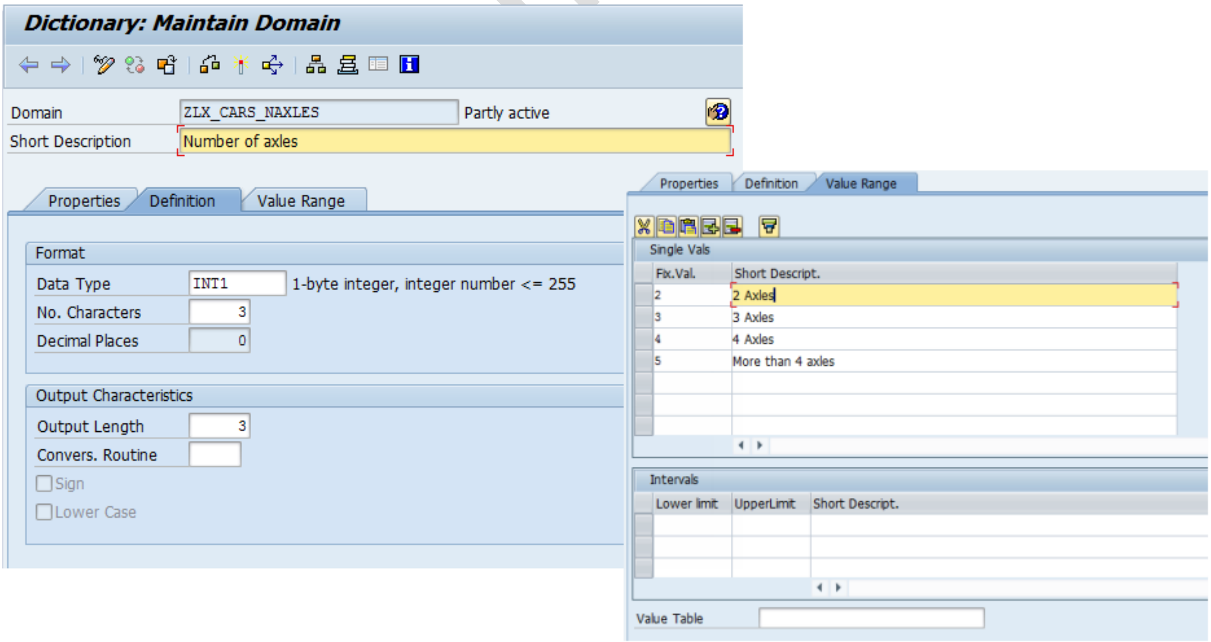
The ABAP Dictionary permits a central description of all the data used in the system without redundancies. New or modified information is automatically provided for all the system components. This ensures data integrity, data consistency and data security. You can create the corresponding objects (tables or views) in the underlying relational database using these data definitions.

Domain

A domain defines a value range. A domain is assigned to a data element. All table fields or structure components that use this data element then have the value range defined by the domain. The relationship between the field or component and the domain is thus defined by the data element of the field or component.
Fields or components that refer to the same domain (with the assigned data elements) are also changed when the domain is changed. This ensures that the value ranges of these fields or components are consistent. Fields or components that are technically the same can thus be combined with a reference to the same domain.
The value range of a domain is defined by specifying a data type and length (and number of decimal places for numeric data types).

How to create a Domain

1. Select object type Domain in the initial screen of the ABAP Dictionary (transaction SE11), enter the name of the domain and choose Create. The maintenance screen for domains appears.
2. Enter an explanatory short text in the field Short Description. You can for example find the domain at a later time using this short text.
3. On the Definition tab page, choose the Data Type, Number of Characters (valid positions without editing characters such as comma or period) and number of Decimal Places (only needed for data types DEC, FLTP, QUAN and CURR).
   Note that some data types have a fixed length. For example, the data type CLNT (client) always has 3 places. If you enter an invalid number of places for such a data type, the system corrects this automatically after issuing a warning.
4. If only certain input values are valid for the domain, you can enter them in the Value Range tab page as fixed values.
5. Save the domain. You are asked to assign the domain to a development package. Indicate your   sub-package created in the previous Unit.
6. Activate the domain.



Dictionary: Maintain Domain

Domain ZLX_CARS_NAXLES Partly active
Short Description Number of axles

Format
Data Type INT1 1-byte integer, integer number <= 255
No. Characters 3
Decimal Places 0

Output Characteristics
Output Length 3
Convers. Routine
□ Sign
□ Lower Case

Single Vals

Fix.Val.	Short Descript.
2	2 Axles
3	3 Axles
4	4 Axles
5	More than 4 axles

Intervals

Lower limit	UpperLimit	Short Descript.

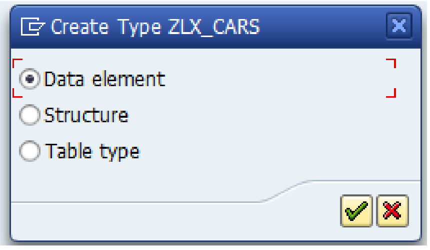
Value Table

Data element

A data element describes either an elementary type or a reference type.
An elementary type is defined by the built-in data type, length and possibly the number of decimal places. These type attributes can either be defined directly in the data element or copied from a domain.
A reference type defines the types of reference variables in ABAP programs.
Information about the meaning of a table field or structure component and information about editing the corresponding screen field can be assigned to a data element. This information is automatically available to all screen fields that refer to the data element.

How to create Data element

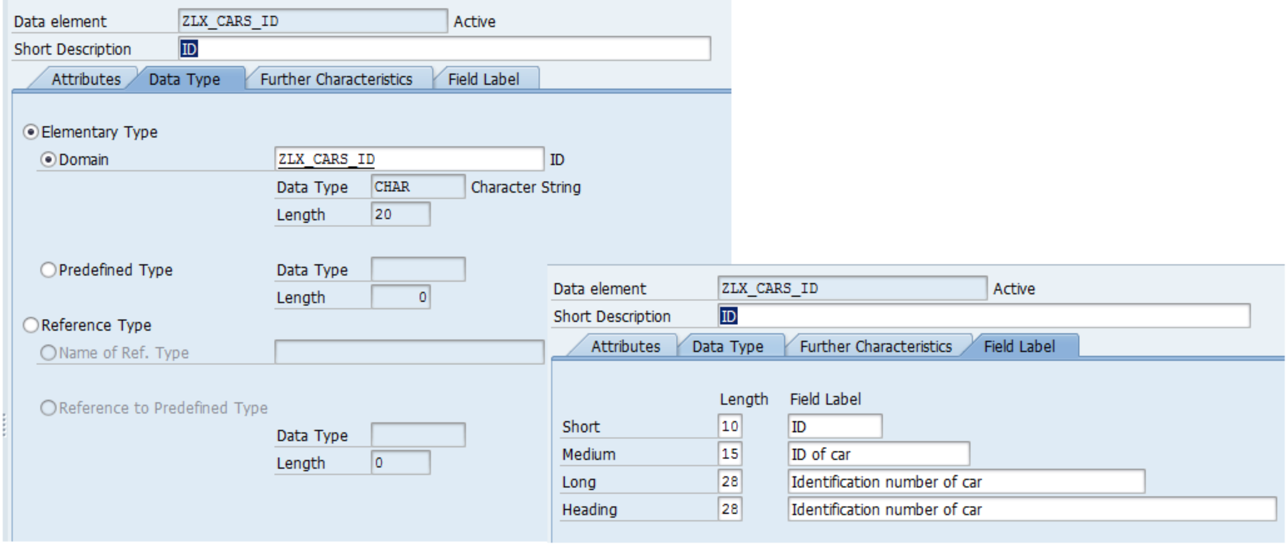
1. In the initial screen of the ABAP Dictionary, select object type Data type, enter the data element name and choose Create. A dialog box appears.



Create Type ZLX_CARS

● Data element
○ Structure
○ Table type

2. Select the data element and choose Continue. The maintenance screen for data elements appears.
3. Enter an explanatory short text. The short text will appear as title in the F1 help for all the screen fields referring to this data element.
4. On the Data Type tab page, define the Data Type, Length and possibly the number of Decimal Places of the data element. You can define these attributes by specifying a domain or by direct type entry.
   If the data element should have the type attributes of a domain, you only have to select Domain and enter the domain name in the corresponding field.
5. On the Field Label tab page you can maintain text information (short, medium, and long field labels and the title) for the data element. You can use this text information in input templates to represent fields that refer to this data element.
6. Save the data element. Assign it to your development package.
7. Activate the data element.



Data element ZLX_CARS_ID Active
Short Description ID

● Elementary Type
● Domain ZLX_CARS_ID ID
Data Type CHAR Character String
Length 20

○ Predefined Type
Data Type
Length 0

○ Reference Type
○ Name of Ref. Type

○ Reference to Predefined Type
Data Type
Length 0

Data element ZLX_CARS_ID Active
Short Description ID

	Length	Field Label
Short	10	ID
Medium	15	ID of car
Long	28	Identification number of car
Heading	28	Identification number of car

Table

Tables can be defined independently of the database in the ABAP Dictionary. The fields of the table are defined with their (database-independent) data types and lengths.

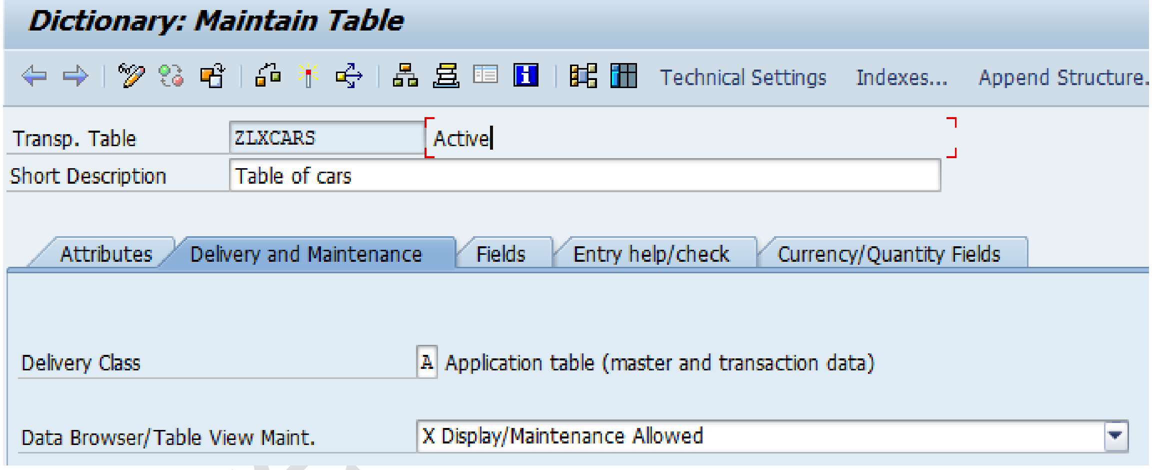
When the table is activated, a physical table definition is created in the database for the table definition stored in the ABAP Dictionary. The table definition is translated from the ABAP Dictionary to a definition of the particular database.

A table definition in the ABAP Dictionary contains the following components:

- Table fields define the field names and data types of the fields contained in the table
- Foreign keys define the relationships between the table and other tables.
- Technical settings control how the table should be created in the database.
- Indexes: To speed up data selection, secondary indexes can be created for the table

How to create a Table

1. In the initial screen of the ABAP Dictionary, select object class Database table, enter the table name and choose Create. The maintenance screen for the table is displayed.
2. Enter an explanatory short text.
3. On the Delivery and Maintenance tab, enter the Delivery Class of the table as **A – Application Table.**
   On this tab, choose an entry from the dropdown box for the Data Browser/Table View Maintenance field. Choose **Display/Maintenance allowed** on this tab page if users with the corresponding authorization may change the data in the table using the Data Browser (transaction SE16).



Dictionary: Maintain Table

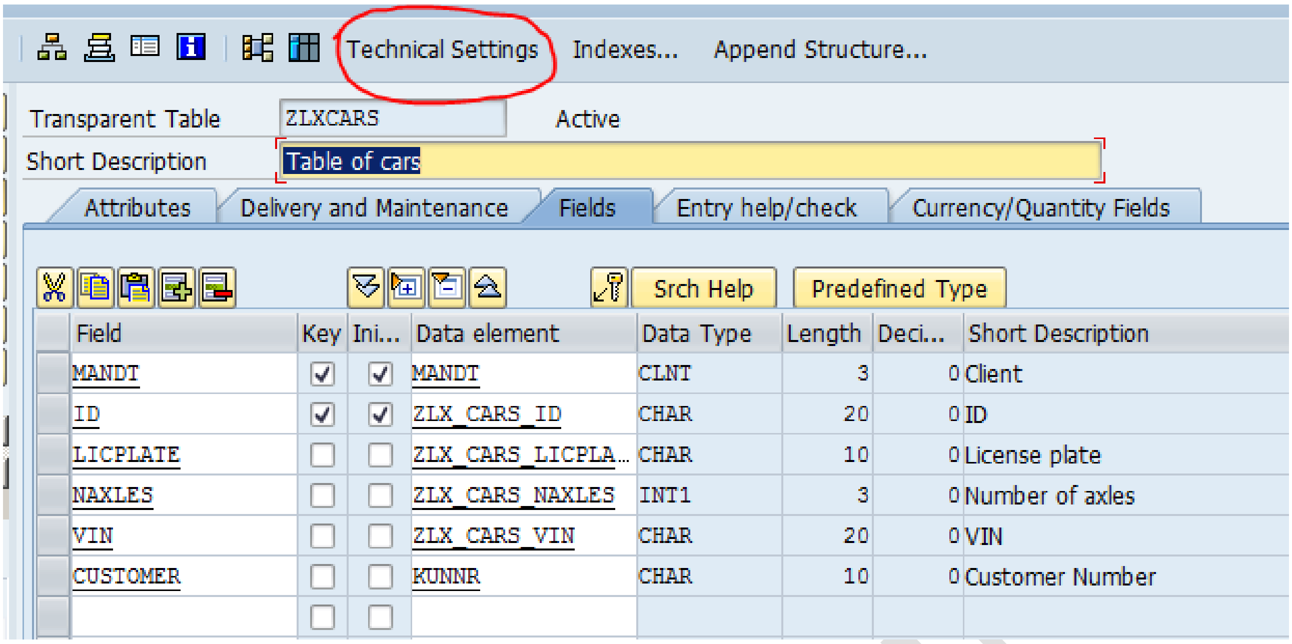
Transp. Table ZLXCARS Active

Short Description Table of cars

Delivery Class A Application table (master and transaction data)

Data Browser/Table View Maint. X Display/Maintenance Allowed

4. On the Fields tab page, enter the table fields. Perform the following steps for each table field:
   a) Enter a name for the table field in the column Field. The field name may only contain letters, digits and underlining, and it must begin with a letter. A field name may not be longer than 16 characters.
   b) Select the Key flag if the field should be part of the table key.
   c) Select the Initial Values flag if a field to be inserted in the database is to be filled with initial values. The initial value used depends on the data type of the field.
   d) Enter the name of a data element in field Data element. In this case the field takes the Data Type, Length, Decimal Places and Short Description from this data element. If there is no suitable data element, you can go to the data element maintenance screen by entering a name and double-clicking.

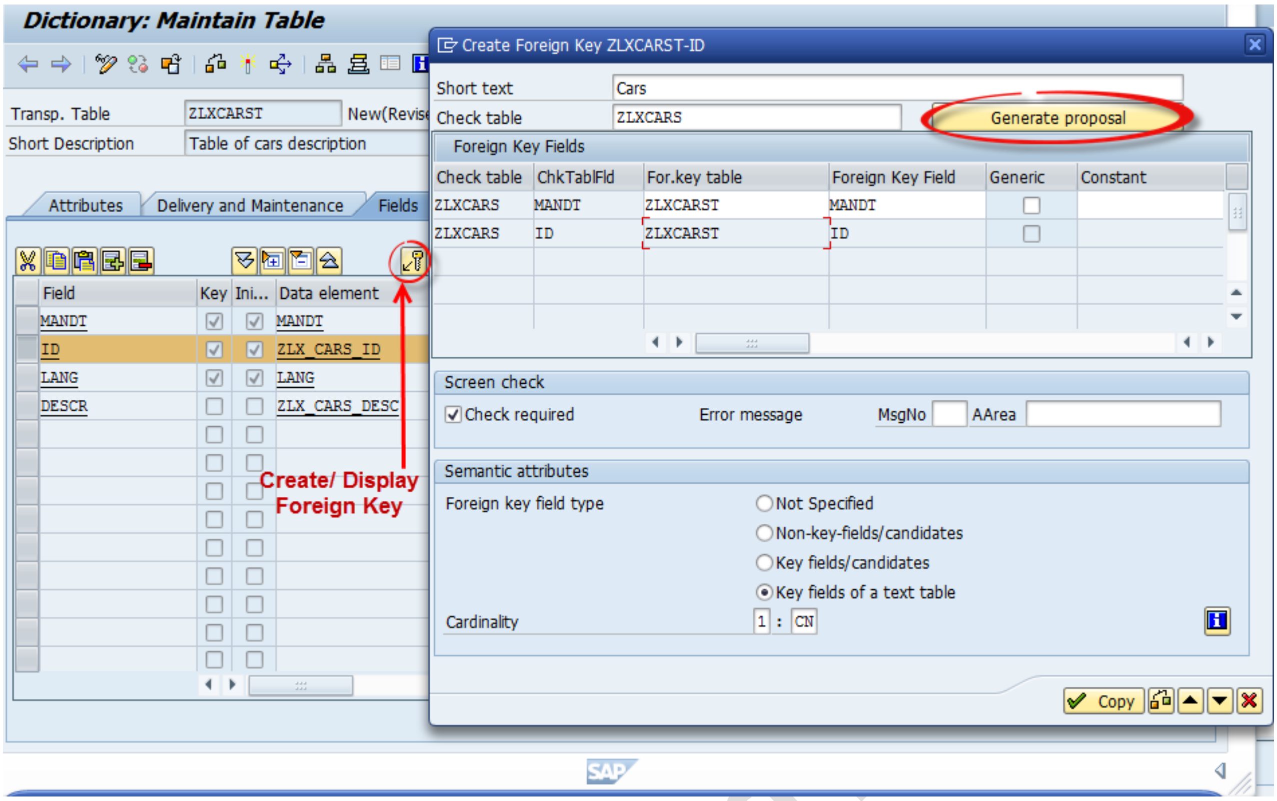


Transparent Table ZLXCARS Active

Short Description Table of cars

Field	Key	Ini...	Data element	Data Type	Length	Deci...	Short Description
MANDT	☑	☑	MANDT	CLNT	3	0	Client
ID	☑	☑	ZLX_CARS_ID	CHAR	20	0	ID
LICPLATE	□	□	ZLX_CARS_LICPLA...	CHAR	10	0	License plate
NAXLES	□	□	ZLX_CARS_NAXLES	INT1	3	0	Number of axles
VIN	□	□	ZLX_CARS_VIN	CHAR	20	0	VIN
CUSTOMER	□	□	KUNNR	CHAR	10	0	Customer Number
	□	□					

5. Maintain the technical settings for the table. The corresponding maintenance screen is displayed with Technical Settings button on the toolbar (or with Goto -> Technical settings). On the Technical Settings maintenance screen specify the Logical storage parameters using F4:
   Data class: APPL0 - Master data, transparent tables
   Size category: 0 - Data records expected: 0 to 5.600
6. Maintain (if necessary) the foreign key relationships of the table to other tables. The purpose of the foreign key is to validate the data that is being entered into a table by checking entries in a check table. Foreign keys are checked by the front end user interface only and it is not checked if you issue a direct a SQL statement to update the database.
   Follow the steps given below to create a foreign key in SAP table:
   a) Select the field of the table for which you want to create the foreign key and press Foreign Keys button.
   b) In the popup window enter an explanatory short text in the field Short text. The short text provides a technical documentation of the meaning of the foreign key.
   c) Enter the Check Table name and press Generate proposal button.
   d) The system proposes the foreign key relation based on the domain. Check that the foreign key relationship proposed by the system is correct and press Copy.
   e) The foreign key is saved and you return to the maintenance screen for the table.



Dictionary: Maintain Table

Transp. Table	ZLXCARST	New(Revis
Short Description	Table of cars description	

Field	Key	Ini...	Data element
MANDT	☑	☑	MANDT
ID	☑	☑	ZLX_CARS_ID
LANG	☑	☑	LANG
DESCR	□	□	ZLX_CARS_DESC

Create/ Display Foreign Key

Create Foreign Key ZLXCARST-ID

Short text	Cars
Check table	ZLXCARS

Foreign Key Fields

Check table	ChkTablFld	For.key table	Foreign Key Field	Generic	Constant
ZLXCARS	MANDT	ZLXCARST	MANDT	□	
ZLXCARS	ID	ZLXCARST	ID	□	

Screen check

☑ Check required   Error message   MsgNo   AArea

Semantic attributes

Foreign key field type
○ Not Specified
○ Non-key-fields/candidates
○ Key fields/candidates
● Key fields of a text table

Cardinality 1 : CN

7. Create (if necessary) secondary indexes for the table. To do this choose Goto -> Indexes.
8. Save the table. Assign it to your development package.
9. Activate the table.
10. You can create entries in your table using transactions SE16n or SE16.

Appendix 1 – Naming Conventions in ABAP DDIC

- Prefix - mandatory!
- DB tables - <prefix>AAA
- Views - <prefix>VAAA
- Domains, Data Elements - <prefix>_AAA
- Structures - <prefix>_S_AAA
- Table types - <prefix>_T_AAA