PUBLIC

How-To Configure Data Replication for MDG Custom Objects (Flex Option)

Applicable Releases:

From MDG 7.0 and from SAP S/4HANA 1511

Version 2.0

July 2023

Document History

Document Version	Description
1.0	First official release of this guide (September 2011)
2.0	Change layout and minor adjustments (July 2023)

1 Business Scenario

SAP Master Data Governance (MDG) provides business processes to find, create, change, and mark master data for deletion. It supports the governance of master data in a central hub and the distribution to connected operational and business intelligence systems.

The processes are workflow-driven and can include several approval and revision phases, and the collaboration of all users participating in the master data maintenance.

MDG offers change request (CR)-based processing of master data with integrated workflow, staging, approval, activation, and distribution.

This how-to guide describes the configuration of data replication for your customer specific data model (flex entity types).

2 Background Information

MDG offers the following options to store active master data (data that has been approved):

- The **reuse option** used by MDG-M and MDG-S stores data in the tables such as `MARA` or `LFA1`.
- The **flex option** used by MDG-F and MDG for Custom Objects stores data in generated tables.

In both options, inactive master data (data that has not yet been approved) is stored in the generated tables.

Data that the MDG system replicates to target systems is always **active data**. The MDG system takes the active data from the SAP ERP tables or from the generated tables depending on the option in use (**reuse option** or **flex option**).

MDG applications such as MDG-M, MDG-S, and MDG-F include standard implementations of the Data Replication Framework (DRF) that read the data and send the messages to the target system. The standard implementations support key mapping and value mapping.

SAP also delivers generic implementations that you can configure to replicate data from customer- specific applications (MDG for Custom Objects). This guide describes the necessary configuration steps.

You can perform most configuration tasks in Customizing for Master Data Governance under SAP Reference IMG > Cross Application Components > Processes and Tools for Enterprise Applications > Master Data Governance.

Additionally, you can use the following transactions:

- `MDGIMG` – IMG Master Data Governance
- `DRFIMG` – IMG Data Replication Framework
- `IDMIMG` – IMG Key Mapping
- `VMIMG` – IMG Value Mapping

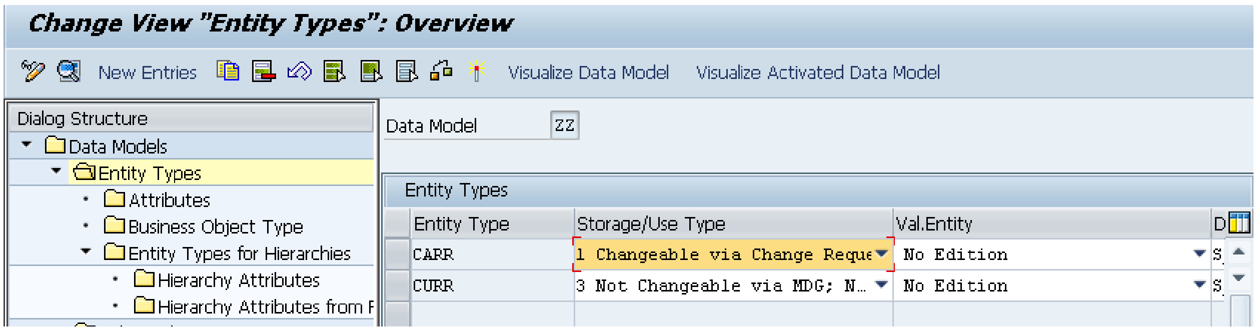
3 Step-by-Step Procedure

3.1 Create the ZZ Data Model Using the Flex Option

This example involves a simple definition of the Data Model that includes two Entity Types and a relationship. The example covers a small part of the `SFLIGHT` scenario, a scenario that is often used in SAP training materials.

The assumption behind the example is that you are building a custom MDG application for creating and governing data about airlines.

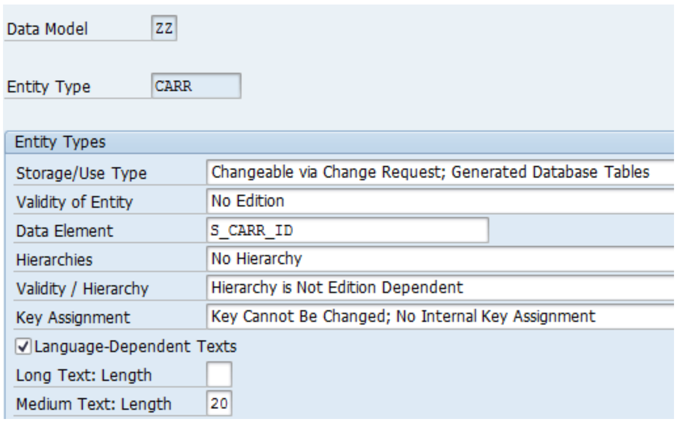
In Customizing for Master Data Governance (transaction `MDGIMG`), choose General Settings > Data Modeling > Edit Data Model. In the Entity Types view, edit the `CARR` Entity Type for Carrier (Airline) as shown below.


Change View "Entity Types": Overview

Data Model: ZZ

Entity Type	Storage/Use Type	Val.Entity
CARR	1 Changeable via Change Reque	No Edition
CURR	3 Not Changeable via MDG; N…	No Edition

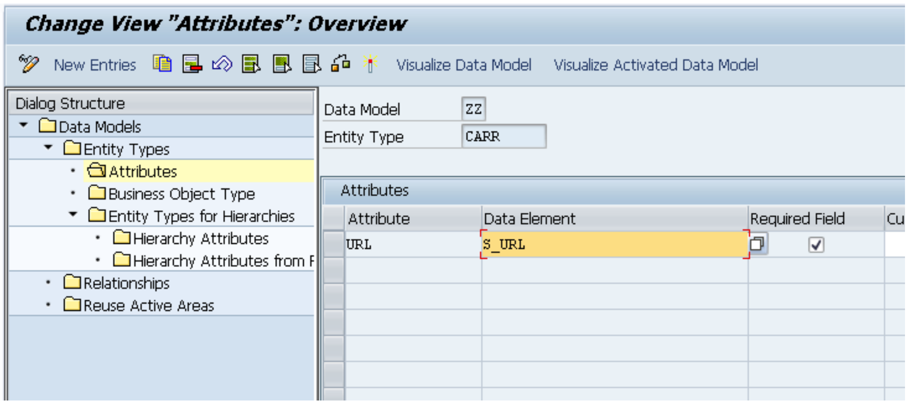
Figure 1 - The Entity Types view of the General Settings > Data Modeling > Edit Data Model Customizing activity



Data Model	ZZ
Entity Type	CARR
Storage/Use Type	Changeable via Change Request; Generated Database Tables
Validity of Entity	No Edition
Data Element	S_CARR_ID
Hierarchies	No Hierarchy
Validity / Hierarchy	Hierarchy is Not Edition Dependent
Key Assignment	Key Cannot Be Changed; No Internal Key Assignment
Language-Dependent Texts	✓
Long Text: Length	
Medium Text: Length	20

Figure 2 - The CARR Entity Type

Specify an attribute `URL` for the `CARR` Entity Type.

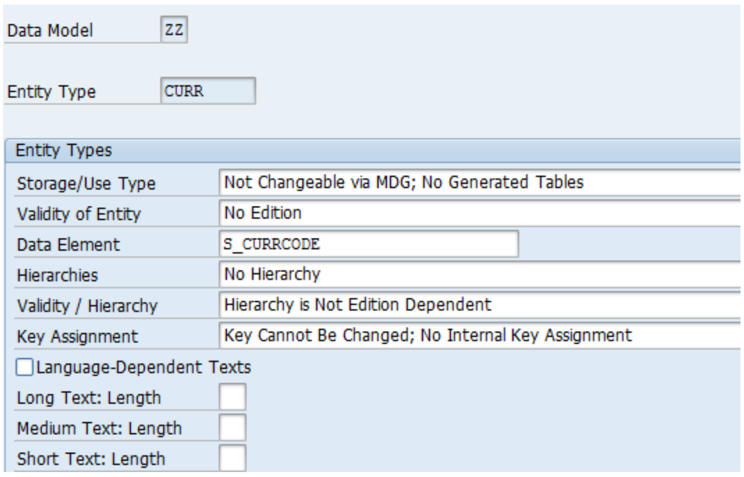

Change View "Attributes": Overview

Data Model: ZZ
Entity Type: CARR

Attribute	Data Element	Required Field
URL	S_URL	✓

Figure 3 - The Attributes view

In the *Entity Types* view, edit the `CURR` Entity Type for *Currency* as shown below.



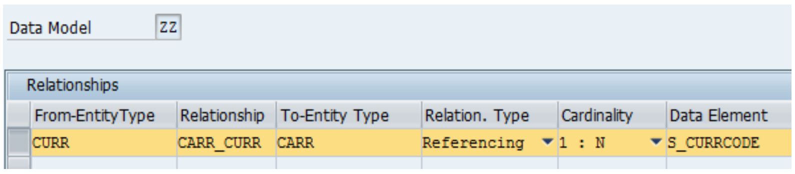
Data Model	ZZ
Entity Type	CURR

Entity Types

Storage/Use Type	Not Changeable via MDG; No Generated Tables
Validity of Entity	No Edition
Data Element	S_CURRCODE
Hierarchies	No Hierarchy
Validity / Hierarchy	Hierarchy is Not Edition Dependent
Key Assignment	Key Cannot Be Changed; No Internal Key Assignment
☐ Language-Dependent Texts	
Long Text: Length	
Medium Text: Length	
Short Text: Length	

Figure 4 - The CURR Entity Type

In the *Relationships* view, specify the relationship between `CARR` and `CURR`.

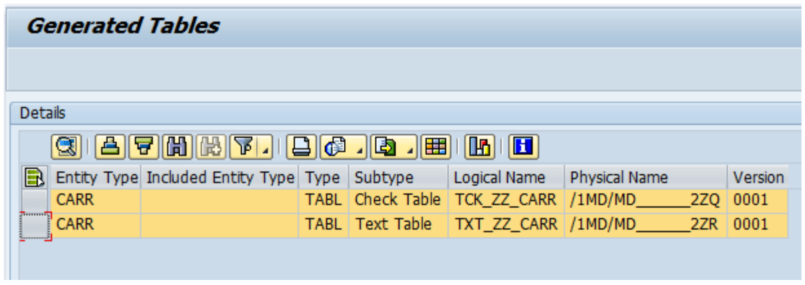


Data Model ZZ

Relationships

From-EntityType	Relationship	To-Entity Type	Relation. Type	Cardinality	Data Element
CURR	CARR_CURR	CARR	Referencing	1 : N	S_CURRCODE

After you generate the Data Model, you can use the `USMD_DATA_MODEL` report to identify the tables generated for your Data Model. You can access this report from transaction `SE38`.



Generated Tables

Details

Entity Type	Included Entity Type	Type	Subtype	Logical Name	Physical Name	Version
CARR		TABL	Check Table	TCK_ZZ_CARR	/1MD/MD______2ZQ	0001
CARR		TABL	Text Table	TXT_ZZ_CARR	/1MD/MD______2ZR	0001

Figure 5 - Generated tables shown after running the USMD_DATA_MODEL report from transaction SE38

3.2 Enable Key Mapping (Optional Step)

This step is required if you want to implement key mapping.

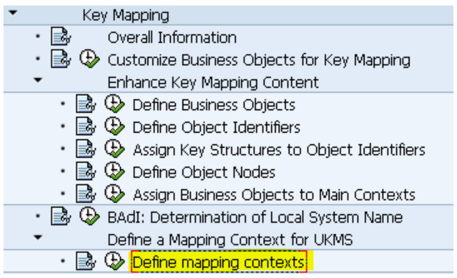
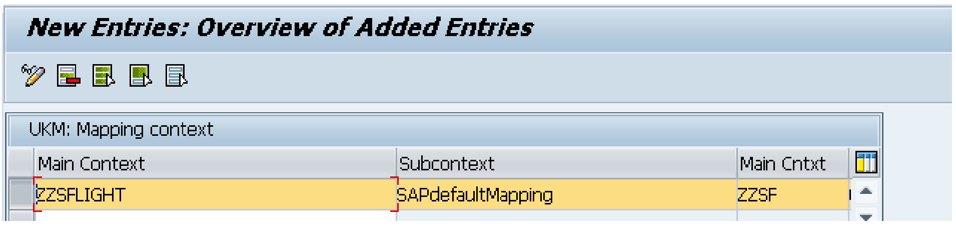

Key Mapping
Overall Information
Customize Business Objects for Key Mapping
Enhance Key Mapping Content
Define Business Objects
Define Object Identifiers
Assign Key Structures to Object Identifiers
Define Object Nodes
Assign Business Objects to Main Contexts
BAdI: Determination of Local System Name
Define a Mapping Context for UKMS
Define mapping contexts

Figure 6 - Transaction IDMIMG: Define a Mapping Context for UKMS -> Define Mapping Contexts

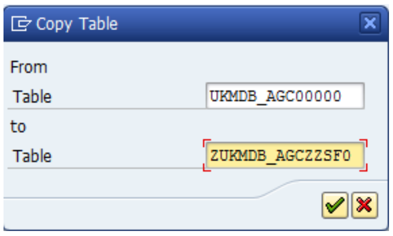

New Entries: Overview of Added Entries

UKM: Mapping context

Main Context	Subcontext	Main Cntxt
ZZSFLIGHT	SAPdefaultMapping	ZZSF

The system generates a set of tables based on standard tables. These tables should have a prefix of

`Z` to identify them as customer specific objects.


Copy Table

From

Table UKMDB_AGC00000

to

Table ZUKMDB_AGCZZSF0

The system requires confirmation of tables to be copied.

3.3 Create a Business Object Type

Data replication always refers to business object types, which are based on data models. You can define business object types in the *Define Business Objects* customizing activity or in the *Define Business Objects Available for Replication* customizing activity.

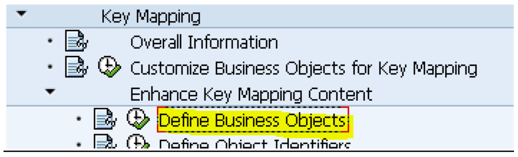

Key Mapping
Overall Information
Customize Business Objects for Key Mapping
Enhance Key Mapping Content
Define Business Objects
Define Object Identifiers

Figure 7 - Transaction IDMIMG

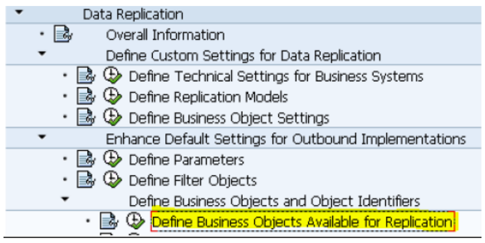
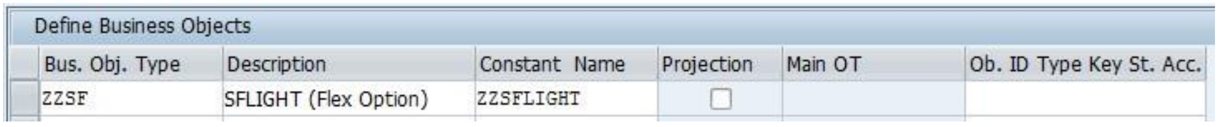
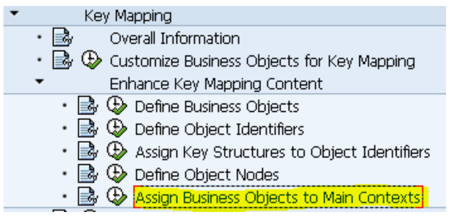

Data Replication
Overall Information
Define Custom Settings for Data Replication
Define Technical Settings for Business Systems
Define Replication Models
Define Business Object Settings
Enhance Default Settings for Outbound Implementations
Define Parameters
Define Filter Objects
Define Business Objects and Object Identifiers
Define Business Objects Available for Replication

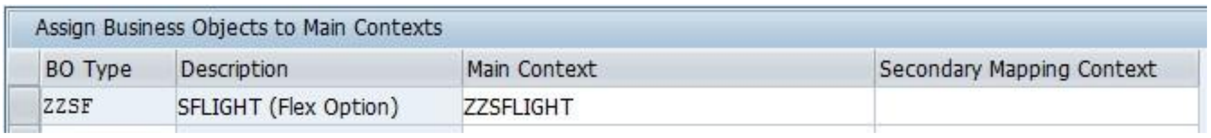
Figure 8 - Transaction DRFIMG


Define Business Objects

Bus. Obj. Type	Description	Constant Name	Projection	Main OT	Ob. ID Type Key St. Acc.
ZZSF	SFLIGHT (Flex Option)	ZZSFLIGHT	☐		

For key mapping, you must assign each Business Object Type to a Main Context. In this example the `ZZSF` Business Object Type is assigned to the `ZZSFLIGHT` Main Context.


Key Mapping
Overall Information
Customize Business Objects for Key Mapping
Enhance Key Mapping Content
Define Business Objects
Define Object Identifiers
Assign Key Structures to Object Identifiers
Define Object Nodes
Assign Business Objects to Main Contexts

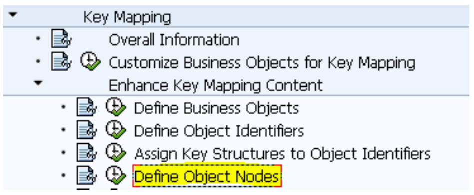


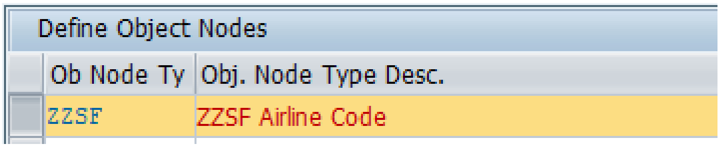
Assign Business Objects to Main Contexts

BO Type	Description	Main Context	Secondary Mapping Context
ZZSF	SFLIGHT (Flex Option)	ZZSFLIGHT	

3.4 Define Object Nodes

Define the `ZZSF` Object Node Type for the `ZZSF` airline code.

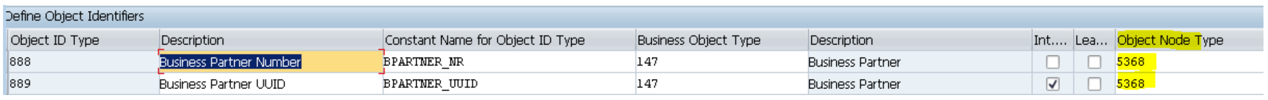

Key Mapping
Overall Information
Customize Business Objects for Key Mapping
Enhance Key Mapping Content
Define Business Objects
Define Object Identifiers
Assign Key Structures to Object Identifiers
Define Object Nodes



Define Object Nodes

Ob Node Ty	Obj. Node Type Desc.
ZZSF	ZZSF Airline Code

You can use the *Define Object Identifiers* customizing activity if there are different Object Identifier Types that must map to each other (for example, if a GUID must map to a number). An example used for the business partner is shown below.



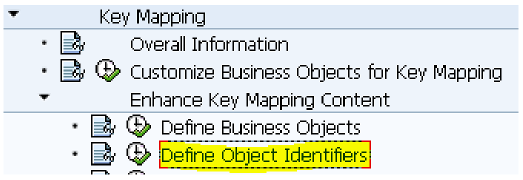
Define Object Identifiers

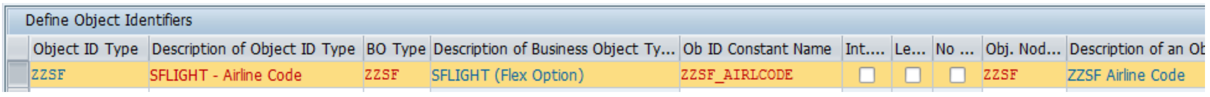
Object ID Type	Description	Constant Name for Object ID Type	Business Object Type	Description	Int....	Lea...	Object Node Type
888	Business Partner Number	BPARTNER_NR	147	Business Partner	☐	☐	5368
889	Business Partner UUID	BPARTNER_UUID	147	Business Partner	✓	☐	5368

Figure 9 - Transaction IDMIMG: Enhance Key Mapping Content -> Define Object Identifiers

3.5 Create and Assign Object Identifier Types

Create the `ZZSF` Object Identifier Type. When doing this, specify the `ZZSF` Object Node Type defined earlier.

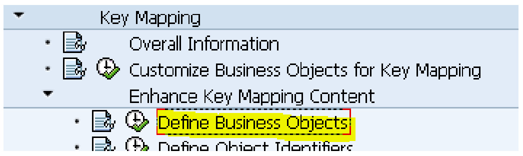

Key Mapping
Overall Information
Customize Business Objects for Key Mapping
Enhance Key Mapping Content
Define Business Objects
Define Object Identifiers

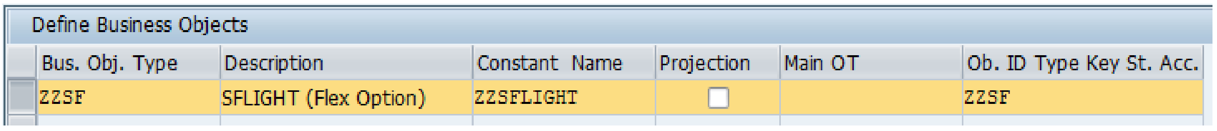


Define Object Identifiers

Object ID Type	Description of Object ID Type	BO Type	Description of Business Object Ty...	Ob ID Constant Name	Int....	Le...	No ...	Obj. Nod...	Description of an Ob
ZZSF	SFLIGHT - Airline Code	ZZSF	SFLIGHT (Flex Option)	ZZSF_AIRLCODE	☐	☐	☐	ZZSF	ZZSF Airline Code

Next, assign the `ZZSF` Object Identifier Type to the `ZZSF` Business Object Type.


Key Mapping
Overall Information
Customize Business Objects for Key Mapping
Enhance Key Mapping Content
Define Business Objects
Define Object Identifiers

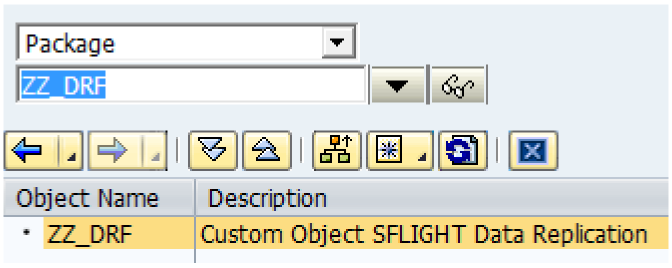


Define Business Objects

Bus. Obj. Type	Description	Constant Name	Projection	Main OT	Ob. ID Type Key St. Acc.
ZZSF	SFLIGHT (Flex Option)	ZZSFLIGHT	☐		ZZSF

3.6 Generate Structures

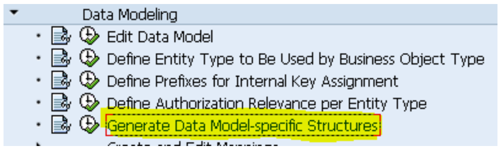
Create a package in SE80.

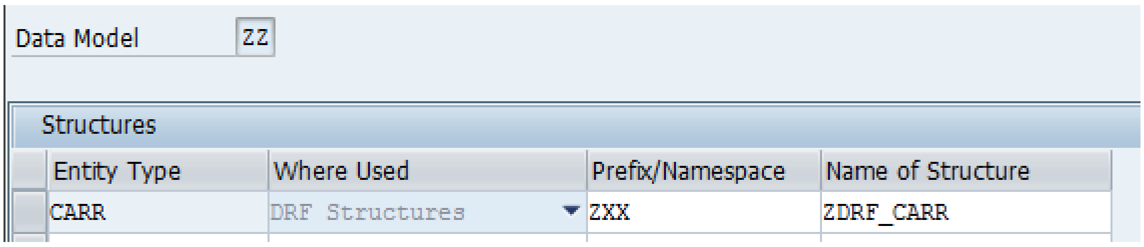

Package

ZZ_DRF

Object Name	Description
ZZ_DRF	Custom Object SFLIGHT Data Replication

Generate data model-specific structures for the CARR Entity Type.

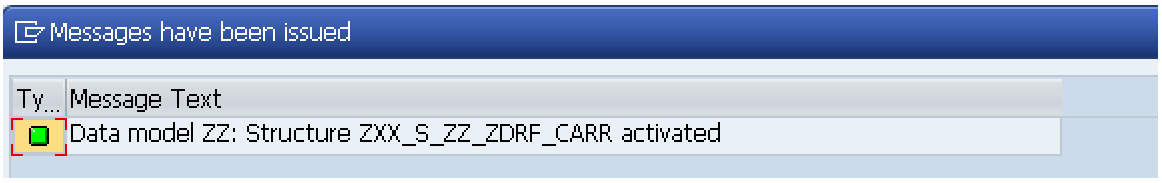

Data Modeling
- Edit Data Model
- Define Entity Type to Be Used by Business Object Type
- Define Prefixes for Internal Key Assignment
- Define Authorization Relevance per Entity Type
- Generate Data Model-specific Structures


Data Model ZZ

Structures

Entity Type	Where Used	Prefix/Namespace	Name of Structure
CARR	DRF Structures	ZXX	ZDRF_CARR

Figure 10 - Elements of the Generate Data Model-specific Structures Customizing activity

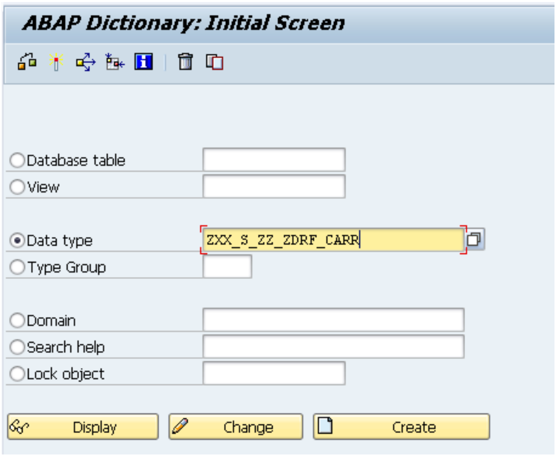

Messages have been issued

Ty...	Message Text
🟩	Data model ZZ: Structure ZXX_S_ZZ_ZDRF_CARR activated

Figure 11 - Confirmation message after the activation of data model-specific structure

Do the same for entity type CURR.

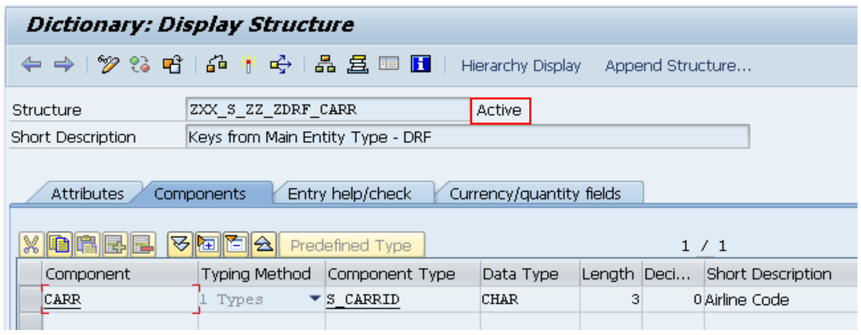
Use transaction SE11 to check the structures for the ZXX_SZZ_ZDRF_CARR data type.


ABAP Dictionary: Initial Screen

Database table	
View	
[✓ Data type]	ZXX_S_ZZ_ZDRF_CARR
Type Group	
Domain	
Search help	
Lock object	

Figure 12 - Searching for data model-specific structure using transaction SE11

Whenever MDG generates these structures, it activates them.



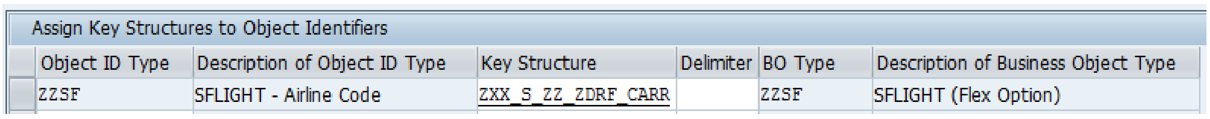
Dictionary: Display Structure

Structure	ZXX_S_ZZ_ZDRF_CARR	Active
Short Description	Keys from Main Entity Type - DRF	

Component	Typing Method	Component Type	Data Type	Length	Deci...	Short Description
CARR	1 Types	S_CARRID	CHAR	3	0	Airline Code

3.7 Assign a Key Structure to Object Identifier Types

Start transaction DRFIMG and navigate to *Data Replication* > *Enhance Default Settings for Outbound Implementations* > *Define Business Objects and Object Identifiers*. Execute customizing activity *Assign Key Structures to Object Identifiers*.

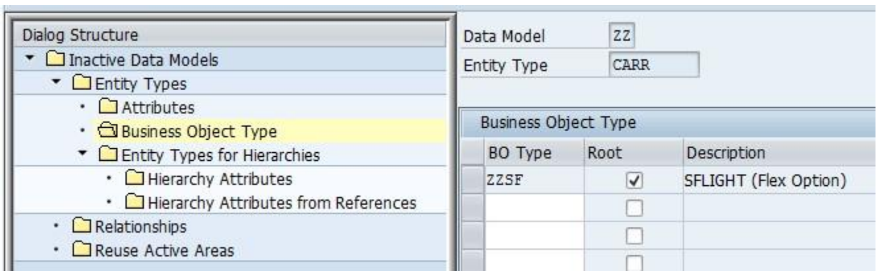


Assign Key Structures to Object Identifiers

Object ID Type	Description of Object ID Type	Key Structure	Delimiter	BO Type	Description of Business Object Type
ZZSF	SFLIGHT - Airline Code	ZXX_S_ZZ_ZDRF_CARR		ZZSF	SFLIGHT (Flex Option)

3.8 Assign a Data Model to a Business Object Type

Start transaction MDGIMG and navigate to *Central Governance* > *General Settings* > *Data Modeling*. Start customizing activity *Edit Data Model*.



Data Model	ZZ
Entity Type	CARR

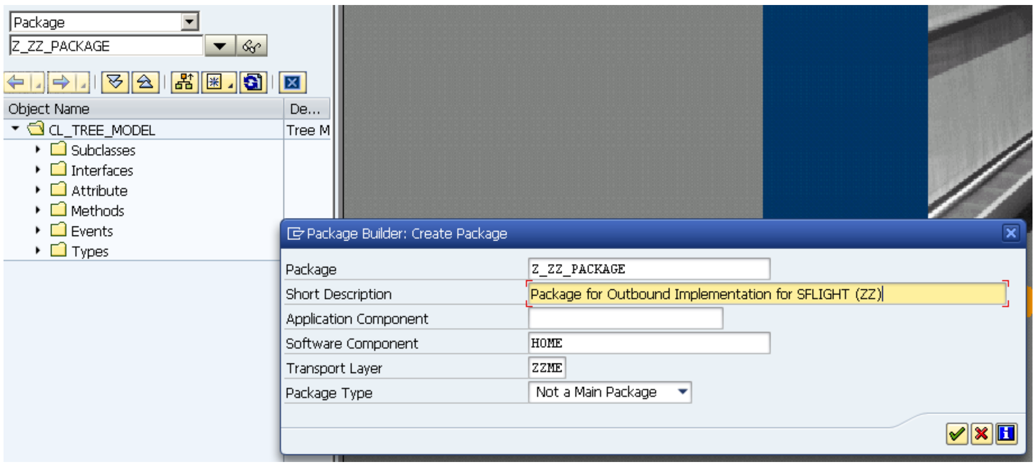
Business Object Type

BO Type	Root	Description
ZZSF	✓	SFLIGHT (Flex Option)

Generate data model ZZ.

3.9 Create an Outbound Interface

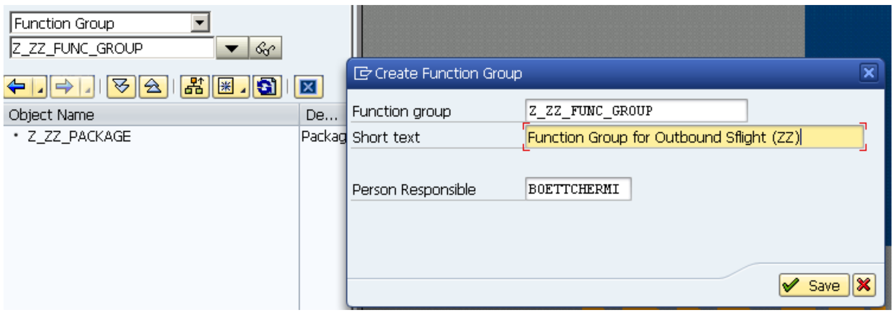
To prepare for the creation of an outbound interface, run transaction SE80 (Create Package / Function Group). Create the `Z_ZZ_PACKAGE` package.



Package Builder: Create Package

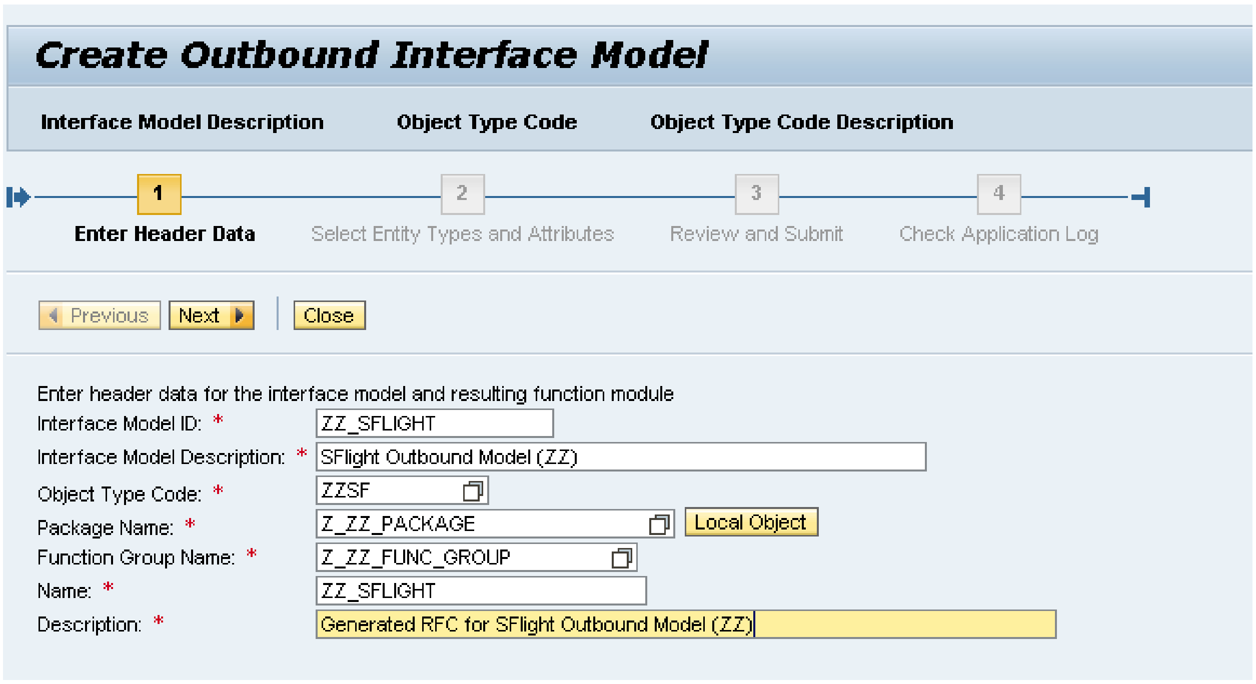
Package	Z_ZZ_PACKAGE
Short Description	Package for Outbound Implementation for SFLIGHT (ZZ)
Application Component	
Software Component	HOME
Transport Layer	ZZME
Package Type	Not a Main Package

Create the `Z_ZZ_FUNC_GROUP` function group in the `Z_ZZ_PACKAGE` package.



Function group: Z_ZZ_FUNC_GROUP
Short text: Function Group for Outbound Sflight (ZZ)
Person Responsible: BOETTCHERMI

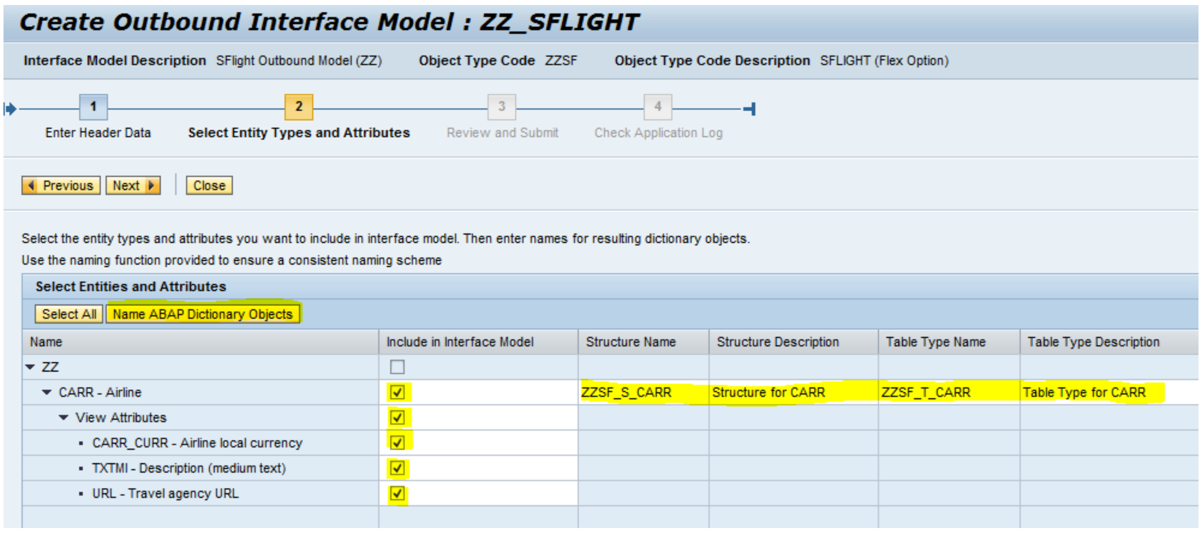
Start transaction `OIF_MAINTAIN`. Create and define an *Interface Model ID*.


Create Outbound Interface Model
Interface Model Description
Object Type Code
Object Type Code Description
1
Enter Header Data
2
Select Entity Types and Attributes
3
Review and Submit
4
Check Application Log
Previous
Next
Close
Local Object

Enter header data for the interface model and resulting function module

Interface Model ID: *	ZZ_SFLIGHT
Interface Model Description: *	SFlight Outbound Model (ZZ)
Object Type Code: *	ZZSF
Package Name: *	Z_ZZ_PACKAGE
Function Group Name: *	Z_ZZ_FUNC_GROUP
Name: *	ZZ_SFLIGHT
Description: *	Generated RFC for SFlight Outbound Model (ZZ)

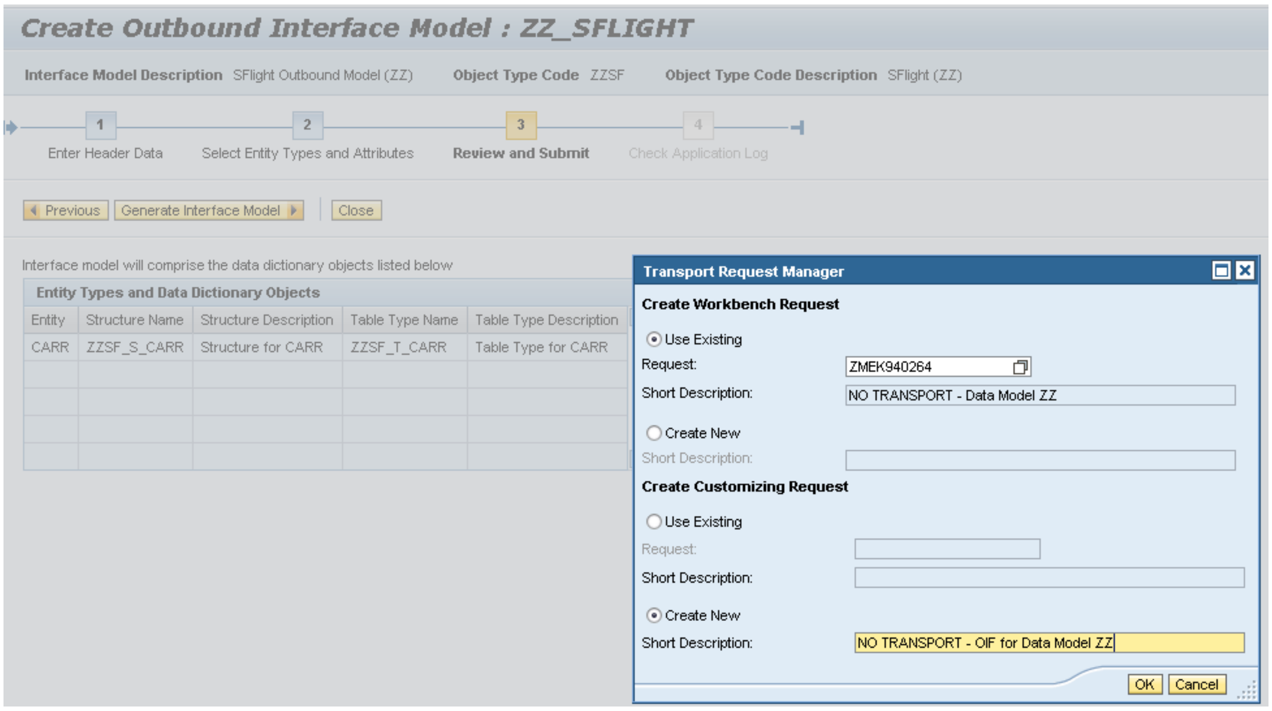
Choose the Name ABAP Dictionary Objects pushbutton to define and name structures.


Create Outbound Interface Model : ZZ_SFLIGHT
1
Enter Header Data
2
Select Entity Types and Attributes
3
Review and Submit
4
Check Application Log
Previous
Next
Close
Select Entities and Attributes
Select All
Name ABAP Dictionary Objects

Interface Model Description SFlight Outbound Model (ZZ) Object Type Code ZZSF Object Type Code Description SFLIGHT (Flex Option)

Select the entity types and attributes you want to include in interface model. Then enter names for resulting dictionary objects.
Use the naming function provided to ensure a consistent naming scheme

Name	Include in Interface Model	Structure Name	Structure Description	Table Type Name	Table Type Description
ZZ	☐				
CARR - Airline	✓	ZZSF_S_CARR	Structure for CARR	ZZSF_T_CARR	Table Type for CARR
View Attributes	✓				
CARR_CURR - Airline local currency	✓				
TXTMI - Description (medium text)	✓				
URL - Travel agency URL	✓				



Create Outbound Interface Model : ZZ_SFLIGHT

Interface Model Description SFlight Outbound Model (ZZ) Object Type Code ZZSF Object Type Code Description SFlight (ZZ)

Interface model will comprise the data dictionary objects listed below

Entity Types and Data Dictionary Objects

Entity	Structure Name	Structure Description	Table Type Name	Table Type Description
CARR	ZZSF_S_CARR	Structure for CARR	ZZSF_T_CARR	Table Type for CARR

Transport Request Manager

Create Workbench Request

(●) Use Existing

Request: ZMEK940264

Short Description: NO TRANSPORT - Data Model ZZ

( ) Create New

Short Description:

Create Customizing Request

( ) Use Existing

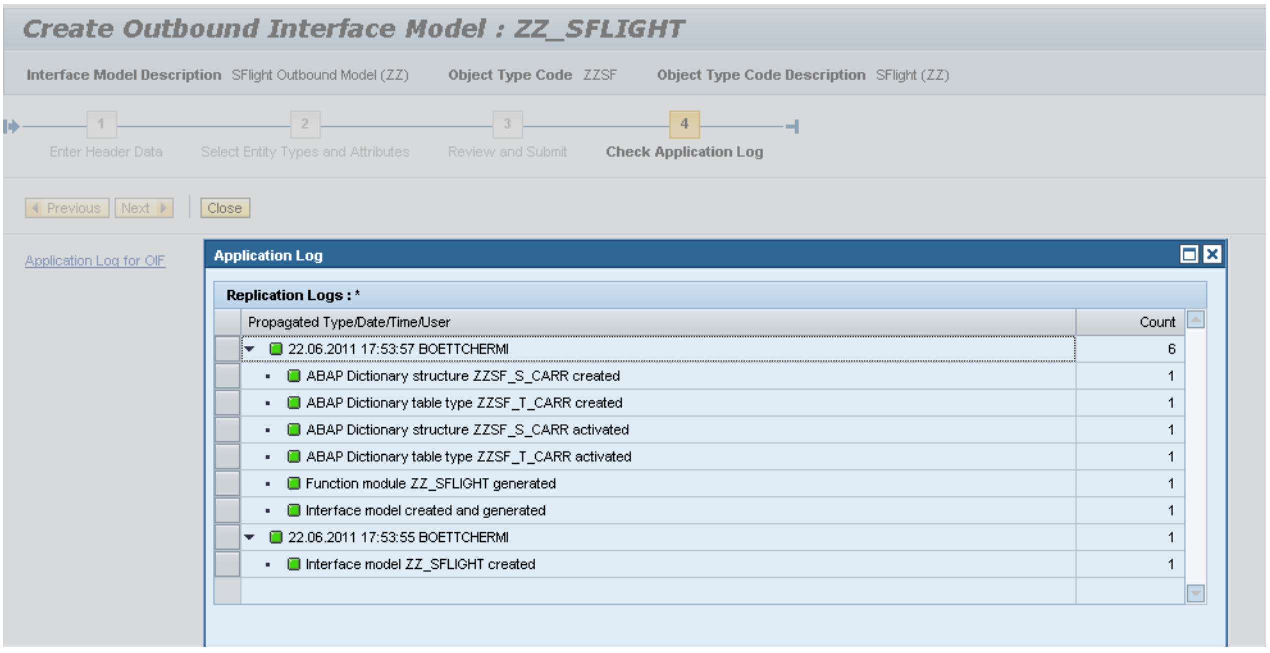
Request:

Short Description:

(●) Create New

Short Description: NO TRANSPORT - OIF for Data Model ZZ

The Transport Request Manager that opens when you complete Step 3 in creating an Outbound Interface Model – Review and Submit. You can use the same transport to transfer the function module in the target system later on.



Create Outbound Interface Model : ZZ_SFLIGHT

Interface Model Description SFlight Outbound Model (ZZ) Object Type Code ZZSF Object Type Code Description SFlight (ZZ)

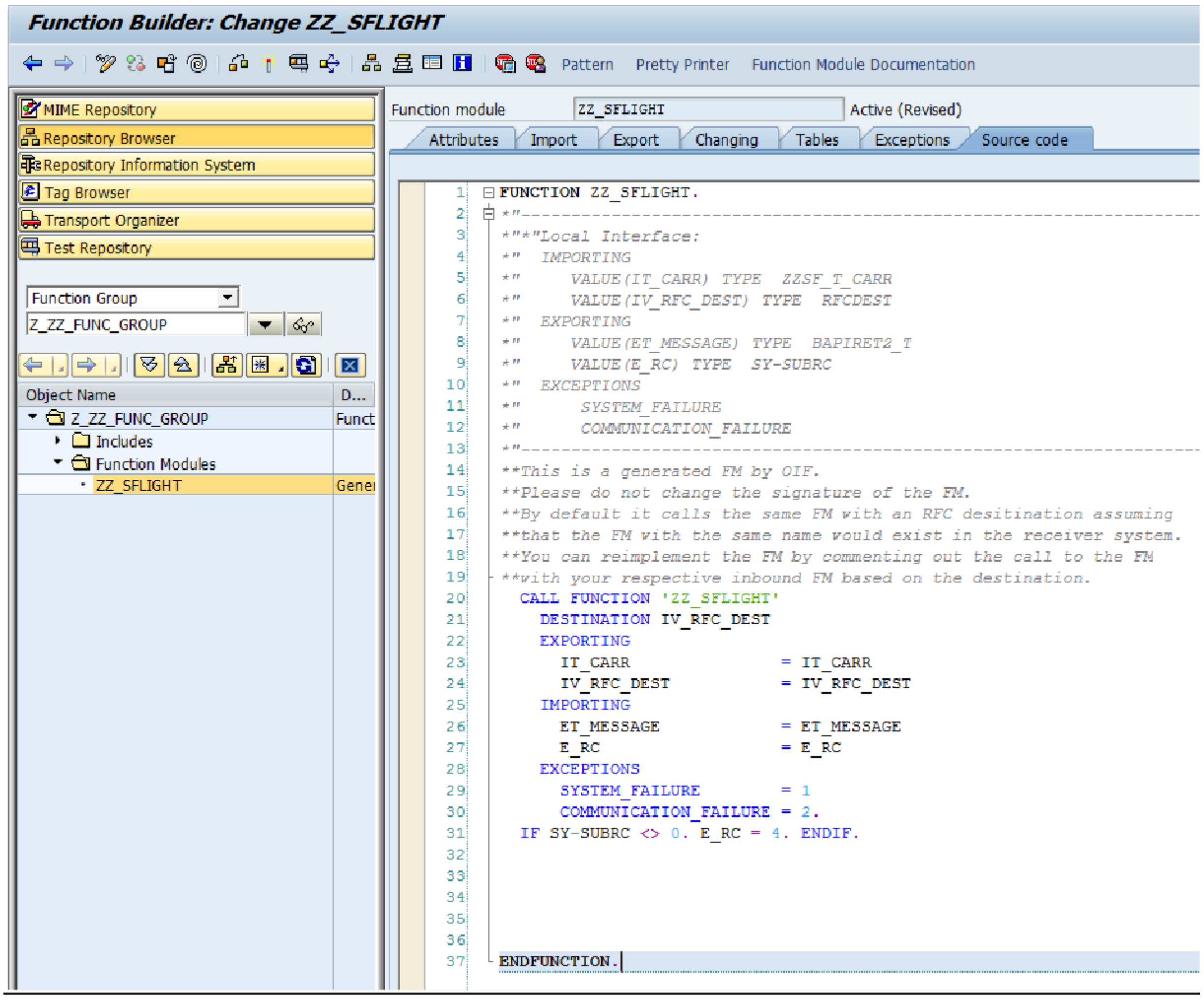
Application Log

Replication Logs : *

Propagated Type/Date/Time/User	Count
🟩 22.06.2011 17:53:57 BOETTCHERMI	6
🟩 ABAP Dictionary structure ZZSF_S_CARR created	1
🟩 ABAP Dictionary table type ZZSF_T_CARR created	1
🟩 ABAP Dictionary structure ZZSF_S_CARR activated	1
🟩 ABAP Dictionary table type ZZSF_T_CARR activated	1
🟩 Function module ZZ_SFLIGHT generated	1
🟩 Interface model created and generated	1
🟩 22.06.2011 17:53:55 BOETTCHERMI	1
🟩 Interface model ZZ_SFLIGHT created	1

The Application Log that opens in Step 4 of creating an Outbound Interface Model – Create Application Log.

After you define the `ZZ_SFLIGHT` Outbound Interface Model, the system generates the `ZZ_SFLIGHT` function module (see below). The outbound implementation defined in the DRF calls this function module.



Function module ZZ_SFLIGHT Active (Revised)

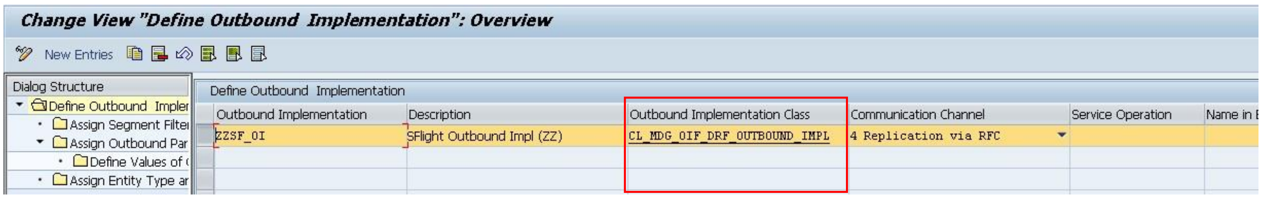
```
FUNCTION ZZ_SFLIGHT.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     VALUE(IT_CARR) TYPE  ZZSF_T_CARR
*"     VALUE(IV_RFC_DEST) TYPE  RFCDEST
*"  EXPORTING
*"     VALUE(ET_MESSAGE) TYPE  BAPIRET2_T
*"     VALUE(E_RC) TYPE  SY-SUBRC
*"  EXCEPTIONS
*"      SYSTEM_FAILURE
*"      COMMUNICATION_FAILURE
*"----------------------------------------------------------------------
**This is a generated FM by OIF.
**Please do not change the signature of the FM.
**By default it calls the same FM with an RFC desitination assuming
**that the FM with the same name would exist in the receiver system.
**You can reimplement the FM by commenting out the call to the FM
**with your respective inbound FM based on the destination.
  CALL FUNCTION 'ZZ_SFLIGHT'
    DESTINATION IV_RFC_DEST
    EXPORTING
      IT_CARR               = IT_CARR
      IV_RFC_DEST           = IV_RFC_DEST
    IMPORTING
      ET_MESSAGE            = ET_MESSAGE
      E_RC                  = E_RC
    EXCEPTIONS
      SYSTEM_FAILURE        = 1
      COMMUNICATION_FAILURE = 2.
  IF SY-SUBRC <> 0. E_RC = 4. ENDIF.





ENDFUNCTION.
```

3.10 Create an Outbound Implementation

Start transaction DRFIMG. Navigate to *Data Replication > Enhance Default Settings for Outbound Implementations*. Execute IMG activity *Define Outbound Implementations*.

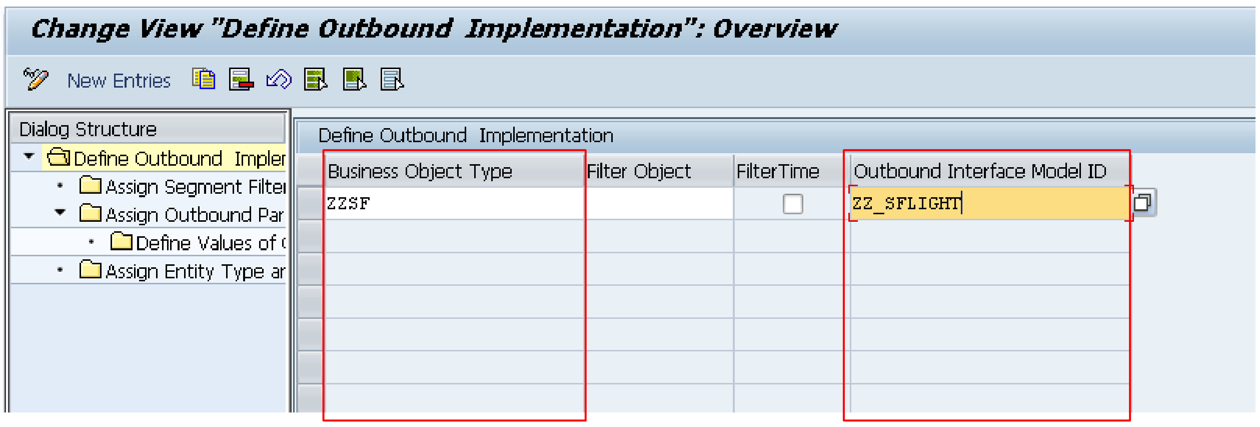
When defining an outbound implementation, use the generic outbound implementation class (`CL_MDG_OIF_DRF_OUTBOUND_IMPL`). You can copy this class to allow additional capabilities that are not supported by default such as key mapping and value mapping. For more information, you can refer to the standard outbound implementations that SAP delivers for other objects.

For your outbound implementation, you must specify a Business Object Type (in this case `ZZSF_01`), an Outbound Implementation Class (in this case `CL_MDG_OIF_DRF_OUTBOUND_IMPL`), and an Outbound Interface Model ID (in this case `ZZ_SFLIGHT`). The *Outbound Interface Model ID* is the last column in a table that requires scrolling. The two screenshots below show the relevant fields.



Define Outbound Implementation

Outbound Implementation	Description	Outbound Implementation Class	Communication Channel	Service Operation	Name in I
ZZSF_OI	SFlight Outbound Impl (ZZ)	CL_MDG_OIF_DRF_OUTBOUND_IMPL	4 Replication via RFC		


Change View "Define Outbound Implementation": Overview

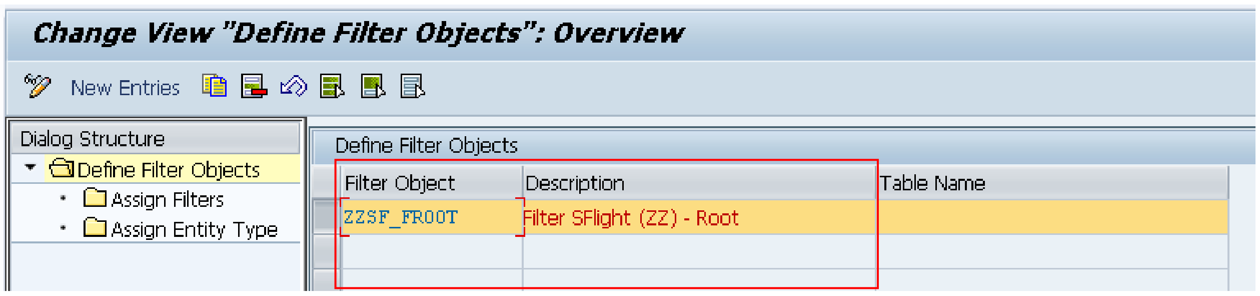
Dialog Structure
- Define Outbound Impler
  - Assign Segment Filte
  - Assign Outbound Par
    - Define Values of
  - Assign Entity Type ar

Define Outbound Implementation

Business Object Type	Filter Object	FilterTime	Outbound Interface Model ID
ZZSF		☐	ZZ_SFLIGHT

3.11 Create a Filter Object

Start transaction DRFIMG. Navigate to *Data Replication* > *Enhance Default Settings for Outbound Implementations*. Execute customizing activity *Define Filter Objects*.


Change View "Define Filter Objects": Overview

Dialog Structure
- Define Filter Objects
  - Assign Filters
  - Assign Entity Type

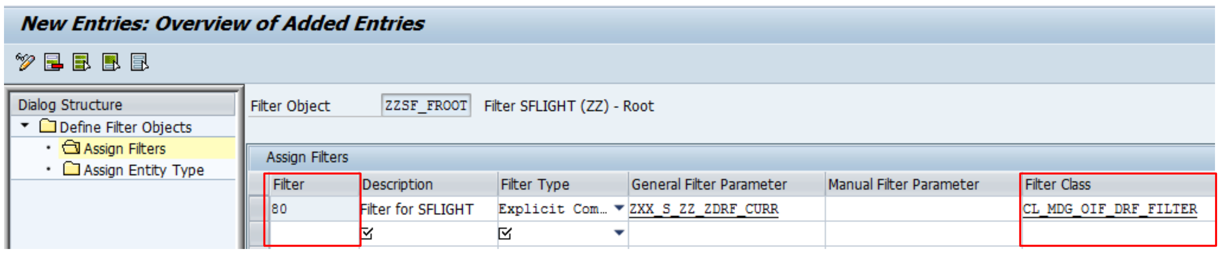
Define Filter Objects

Filter Object	Description	Table Name
ZZSF_FROOT	Filter SFlight (ZZ) - Root	

Leave field *Table Name* blank. A complex filter such as the one in the example does not require a table name. The system only requires table names for simple filters. Such filters are only available for standard applications that are built using the reuse option.

If required, you can define your own structure to include all relevant fields from the generated table. In the *Assign Filters* view, apply the following settings.

- For field *Filter* use codes between `80` and `99`. This range is assigned to the customer namespace.
- Use the generic Filter Class `CL_MDG_OIF_DRF_FILTER`.


New Entries: Overview of Added Entries

Dialog Structure
- Define Filter Objects
  - Assign Filters
  - Assign Entity Type

Filter Object ZZSF_FROOT Filter SFLIGHT (ZZ) - Root

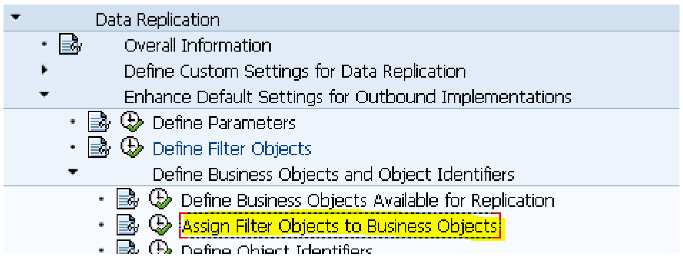
Assign Filters

Filter	Description	Filter Type	General Filter Parameter	Manual Filter Parameter	Filter Class
80	Filter for SFLIGHT	Explicit Com…	ZXX_S_ZZ_ZDRF_CURR		CL_MDG_OIF_DRF_FILTER

You do not have to assign an Entity Type.

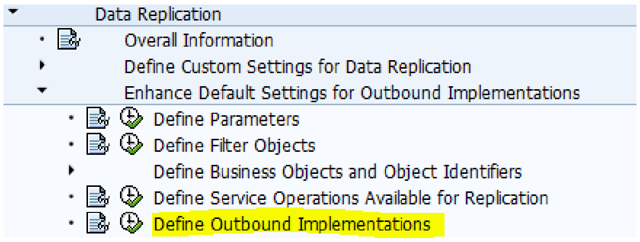
3.12 Assign a Filter to a Business Object Type or an Outbound Implementation

You can assign a Filter Object to a Business Object Type.

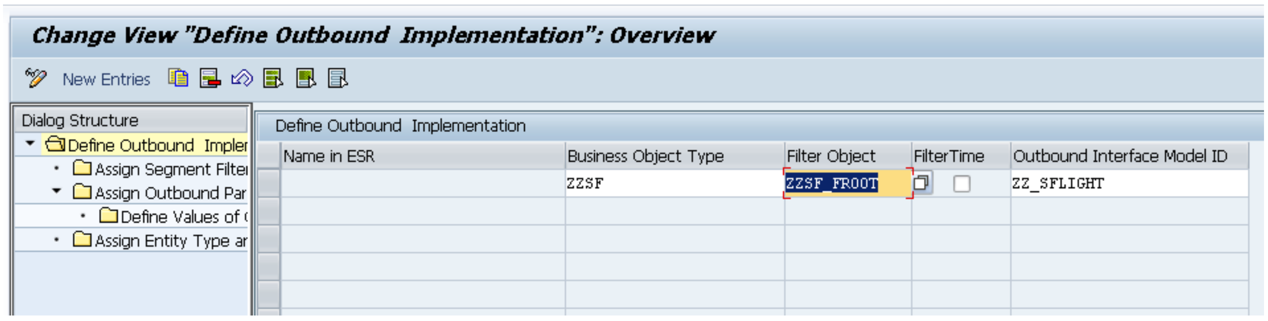


- Data Replication
  - Overall Information
  - Define Custom Settings for Data Replication
  - Enhance Default Settings for Outbound Implementations
    - Define Parameters
    - Define Filter Objects
    - Define Business Objects and Object Identifiers
      - Define Business Objects Available for Replication
      - Assign Filter Objects to Business Objects
      - Define Object Identifiers

Alternatively, you can assign a Filter Object to an outbound implementation, which is more specific than a Business Object Type.



- Data Replication
  - Overall Information
  - Define Custom Settings for Data Replication
  - Enhance Default Settings for Outbound Implementations
    - Define Parameters
    - Define Filter Objects
    - Define Business Objects and Object Identifiers
    - Define Service Operations Available for Replication
    - Define Outbound Implementations



Change View "Define Outbound Implementation": Overview

New Entries

Dialog Structure
- Define Outbound Impler
  - Assign Segment Filter
  - Assign Outbound Par
    - Define Values of
  - Assign Entity Type ar

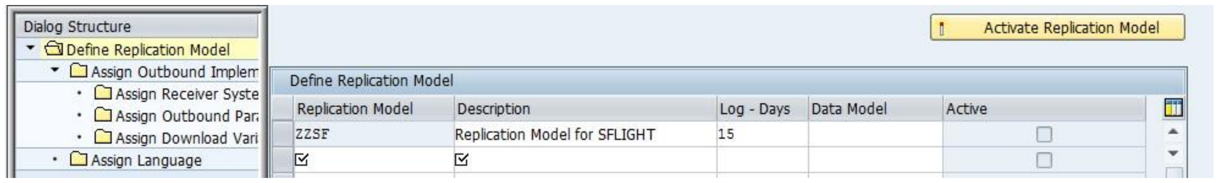
Define Outbound Implementation

Name in ESR	Business Object Type	Filter Object	FilterTime	Outbound Interface Model ID
	ZZSF	ZZSF_FROOT	☐	ZZ_SFLIGHT

3.13 Create a Replication Model

Define the ZZSF Replication Model. This customizing activity is client-specific.

Run transaction DRFIMG and choose **Define Custom Settings for Outbound Implementations > Define Replication Models**.



Dialog Structure
- Define Replication Model
  - Assign Outbound Implem
    - Assign Receiver Syste
    - Assign Outbound Para
    - Assign Download Vari
  - Assign Language

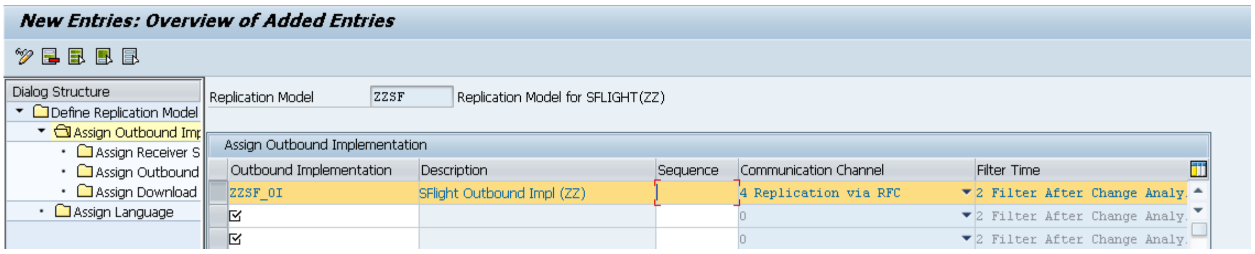
Activate Replication Model

Define Replication Model

Replication Model	Description	Log - Days	Data Model	Active
ZZSF	Replication Model for SFLIGHT	15		☐
				☐

The *Data Model* field is specific to MDG-Financials. You do not need to specify a data model for custom objects.

Assign the ZZSF_01 outbound implementation to the ZZSF Replication Model.



New Entries: Overview of Added Entries

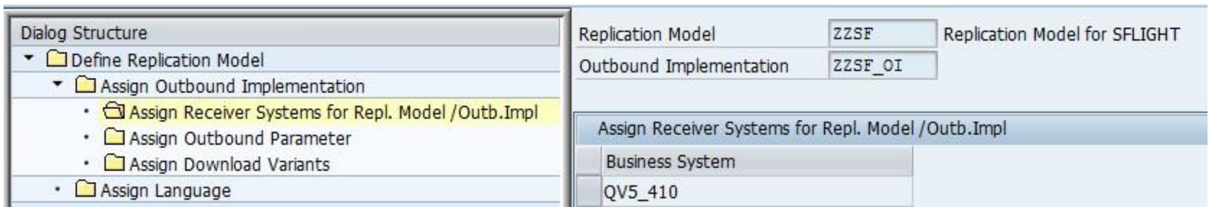
Dialog Structure
- Define Replication Model
  - Assign Outbound Imp
    - Assign Receiver S
    - Assign Outbound
    - Assign Download
  - Assign Language

Replication Model ZZSF Replication Model for SFLIGHT(ZZ)

Assign Outbound Implementation

Outbound Implementation	Description	Sequence	Communication Channel	Filter Time
ZZSF_OI	SFlight Outbound Impl (ZZ)		4 Replication via RFC	2 Filter After Change Analy.
			0	2 Filter After Change Analy.
			0	2 Filter After Change Analy.

Assign the Business System or systems that act as receiver systems for the combination of the Replication Model and the Outbound Implementation.



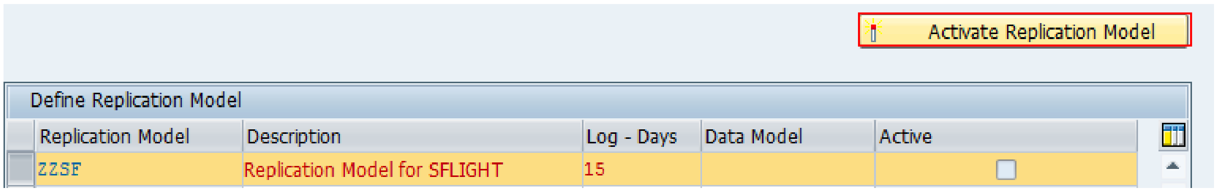
Dialog Structure
- Define Replication Model
  - Assign Outbound Implementation
    - Assign Receiver Systems for Repl. Model /Outb.Impl
    - Assign Outbound Parameter
    - Assign Download Variants
  - Assign Language

Replication Model	ZZSF	Replication Model for SFLIGHT
Outbound Implementation	ZZSF_OI	

Assign Receiver Systems for Repl. Model /Outb.Impl

Business System
QV5_410

Make sure you activate the Replication Model.

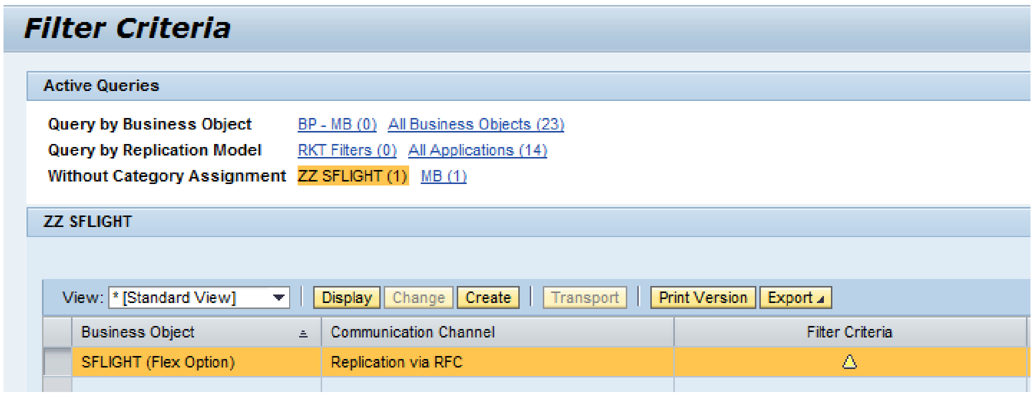


Define Replication Model

Replication Model	Description	Log - Days	Data Model	Active
ZZSF	Replication Model for SFLIGHT	15		☐

3.14 Create a Filter

Run transaction DRFF.



Filter Criteria

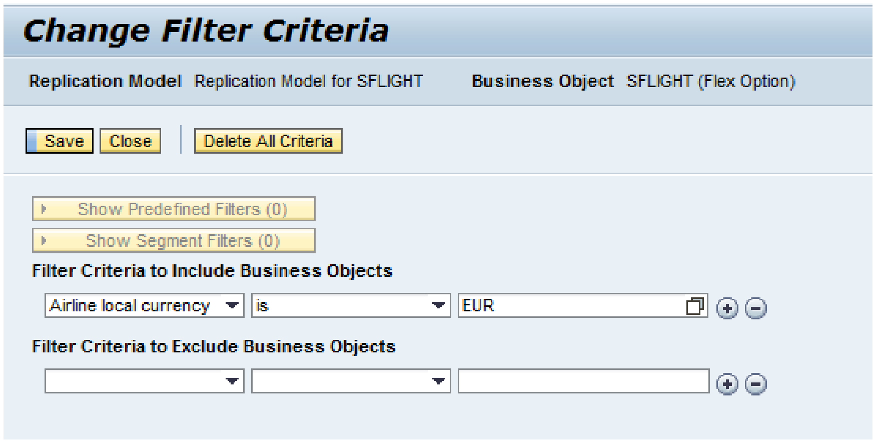
Active Queries

Query by Business Object	BP - MB (0)	All Business Objects (23)
Query by Replication Model	RKT Filters (0)	All Applications (14)
Without Category Assignment	ZZ SFLIGHT (1)	MB (1)

ZZ SFLIGHT

Business Object	Communication Channel	Filter Criteria
SFLIGHT (Flex Option)	Replication via RFC	△

Define a filter specifying that Airline local currency is EUR.



Change Filter Criteria

Replication Model Replication Model for SFLIGHT Business Object SFLIGHT (Flex Option)

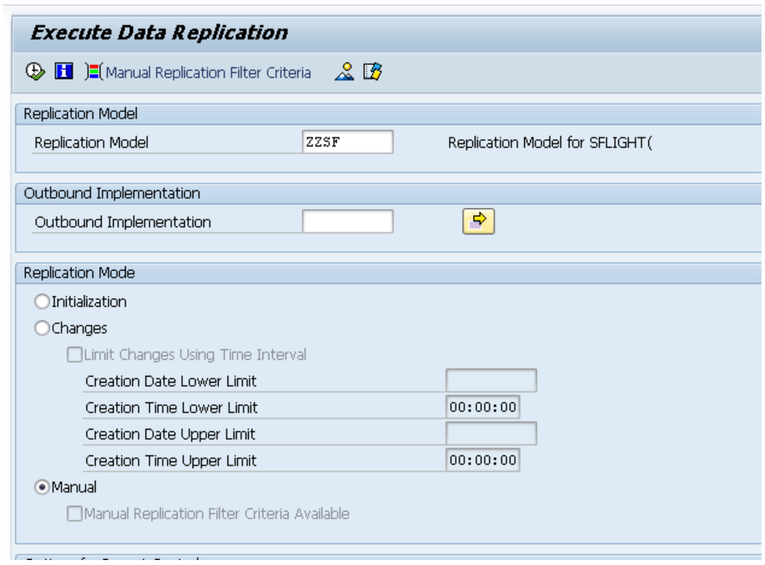
Filter Criteria to Include Business Objects

Airline local currency	is	EUR

Filter Criteria to Exclude Business Objects

3.15 Replicate Data

Execute data replication in transaction `DRFOUT`. Enter Replication Model ZZSF and execute the program.



Execute Data Replication

Replication Model

Replication Model	ZZSF	Replication Model for SFLIGHT(

Outbound Implementation

Outbound Implementation	

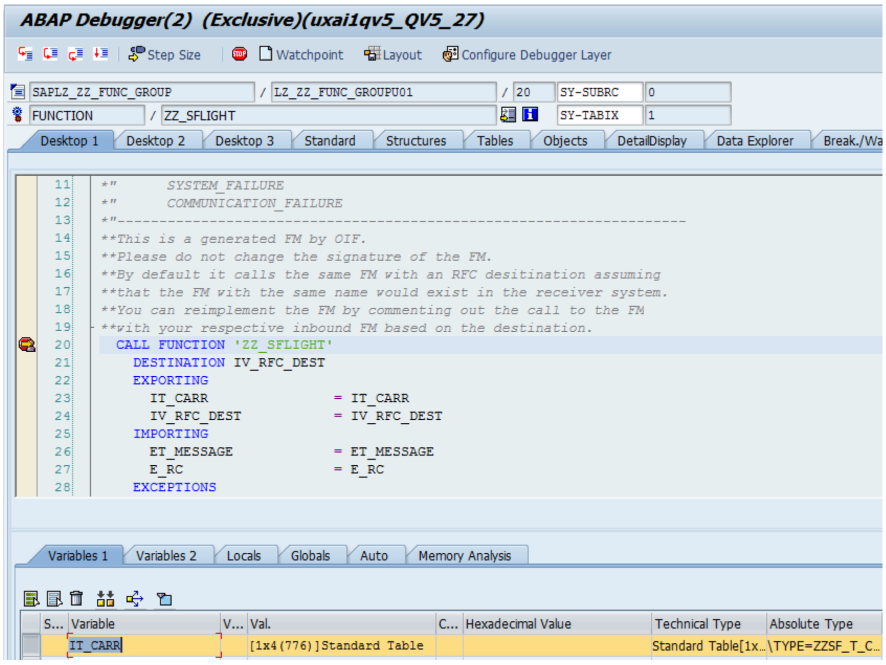
Replication Mode

- ( ) Initialization
- ( ) Changes
  - [ ] Limit Changes Using Time Interval

Creation Date Lower Limit	
Creation Time Lower Limit	00:00:00
Creation Date Upper Limit	
Creation Time Upper Limit	00:00:00

- [✓ Manual]
  - [ ] Manual Replication Filter Criteria Available

The system calls the ZZ_SFLIGHT outbound interface.



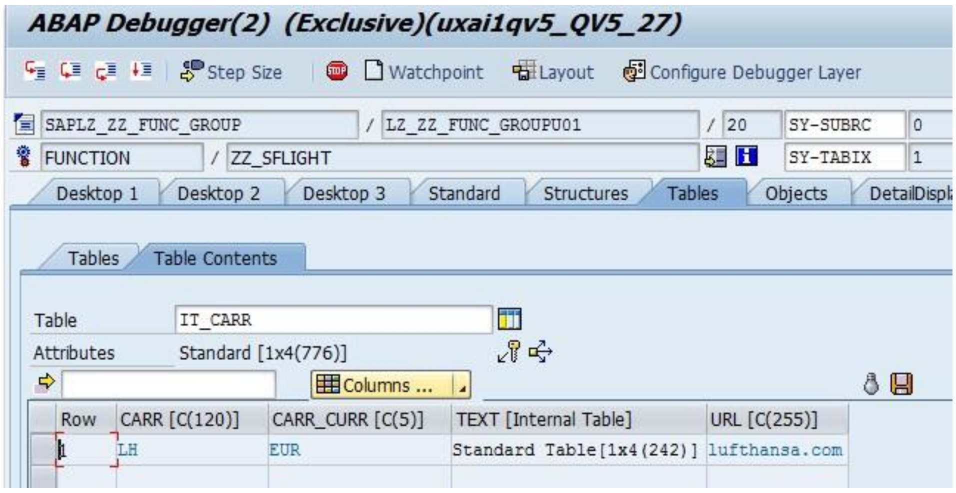
ABAP Debugger(2) (Exclusive)(uxai1qv5_QV5_27)

SAPLZ_ZZ_FUNC_GROUP	/ LZ_ZZ_FUNC_GROUPU01	/ 20	SY-SUBRC 0
FUNCTION	/ ZZ_SFLIGHT		SY-TABIX 1

```
*"      SYSTEM_FAILURE
*"      COMMUNICATION_FAILURE
*"----------------------------------------------------------------------
**This is a generated FM by OIF.
**Please do not change the signature of the FM.
**By default it calls the same FM with an RFC desitination assuming
**that the FM with the same name would exist in the receiver system.
**You can reimplement the FM by commenting out the call to the FM
**with your respective inbound FM based on the destination.
  CALL FUNCTION 'ZZ_SFLIGHT'
    DESTINATION IV_RFC_DEST
    EXPORTING
      IT_CARR                = IT_CARR
      IV_RFC_DEST            = IV_RFC_DEST
    IMPORTING
      ET_MESSAGE             = ET_MESSAGE
      E_RC                   = E_RC
    EXCEPTIONS
```

S...	Variable	V...	Val.	C...	Hexadecimal Value	Technical Type	Absolute Type
	IT_CARR		[1x4(776)]Standard Table			Standard Table[1x…	\TYPE=ZZSF_T_C…

Objects specified as filter criteria are passed to the ZZ_SFLIGHT function module.



ABAP Debugger(2) (Exclusive)(uxai1qv5_QV5_27)

SAPLZ_ZZ_FUNC_GROUP	/ LZ_ZZ_FUNC_GROUPU01	/ 20	SY-SUBRC 0
FUNCTION	/ ZZ_SFLIGHT		SY-TABIX 1

Table	IT_CARR
Attributes	Standard [1x4(776)]

Row	CARR [C(120)]	CARR_CURR [C(5)]	TEXT [Internal Table]	URL [C(255)]
1	LH	EUR	Standard Table[1x4(242)]	lufthansa.com

Note, this example ends with the data passed to the function module in the sender system. The next step is to create a function module with the same name (`ZZ_SFLIGHT`) in the receiver system. The function module in the sender system (shown above) calls the function module in the receiver system. Additionally, you need to implement settings that allow the posting of the data.

You can consider more sophisticated implementations like using ALE / IDocs or WebServices for the data replication. To do this, start by implementing an RFC-enabled function module in the receiver system. Based on the implementation of the function module, you can generate a BAPI and an IDoc (including functions modules for sending and receiving). The function module of the sender system (shown above) calls the function module you implement in the receiver system. Likewise, you can generate and consume a WebService.

4 Additional Information

4.1 Further Reading

4.1.1 Information on SAP MDG on SAP S/4HANA

- Exchange knowledge: SAP Community | Q&A | Blog
- Try SAP Master Data Governance on S/4HANA for free: Trial Version
- Learn more: Latest Release | Webinars | Help Portal | How-to Information | Key Presentations

4.1.2 SAP Roadmap Explorer

- Please see the roadmap for SAP Master Data Governance

4.1.3 Related Information

- Learn more: Floorplan Manager for Web Dynpro ABAP | How to Adapt FPM | FPM Blog | How-to Information | Service Mapping Tool | SAP S/4HANA Cookbook CVI