

## Chapter 35: Checking User Authorizations

### Overview

Much of the data in an R/3 system has to be protected so that unauthorized users cannot access it. Therefore the appropriate authorization is required before a user can carry out certain actions in the system. When you log on to the R/3 system, the system checks in the user master record to see which transactions you are authorized to use. An authorization check is implemented for every sensitive transaction.

If you wish to protect a transaction that you have programmed yourself, then you must implement an authorization check. This means you have to:

- allocate an authorization object in the definition of the transaction;
- program an AUTHORITY-CHECK.

```
AUTHORITY-CHECK OBJECT <authorization object>
  ID <authority-field1> FIELD <field-value1>
  ID <authority-field2 > FIELD <field-value2>
  ...
  ID <authority-fieldn> FIELD <field-valuen>.
```

The **OBJECT** parameter specifies the authorization object.

The **ID** parameter specifies an authorization field (in the authorization object).

The **FIELD** parameter specifies a value for the authorization field.

The authorization object and its fields have to be suitable for the transaction. In most cases you will be able to use the existing authorization objects to protect your data. But new developments may require that you define new authorization objects and fields.

The following topics provide more information:

*Defining an Authority Check*

*Defining Authorization Objects*

*Defining Authorization Fields*

### Contents

<b>Defining an Authority Check .....</b>	<b>35-2</b>
Defining Authorization Objects .....	35-4
Defining Authorization Fields.....	35-4

## Defining an Authority Check

As an example of handling authority checks, start with transaction tz80, one of the example transactions delivered with each system.

This example from the Flight Reservation system consists of two screens. In the first screen the user can enter flight identifiers and request flight details (by pressing the *Display* pushbutton) or press the *Change* pushbutton to change the flight data.

The S\_CARRID authorization object is allocated to the transaction tz80. This authorization object contains two fields. Generic values can be entered in the first field *Airline carrier*. In the second field *Activity* you can select between create(01), change(02) and display(03).

Authorization fields	
CARRID	Airline carrier Id
ACTUT	Activity

When you have programmed a new transaction, you can specify an authorization object in the definition of the transaction code. Next to the entry field for the authorization

object is a pushbutton labelled *Values*. You use this pushbutton to enter the field values for the authorization object that should be checked when the transaction is started.

Transaction code: T281

Transaction text: Authorizations

Program: SAPMTZ80

Screen number: 100

Authorization object: S\_CARRID

Values

Fields	Values
ACTVT	03
CARRID	*

A user wanting to call transaction tz80 needs an authorization in his user master record for the S\_CARRID authorization object. It has to contain the value *display* (03) for the ACTVT field and a value for the *Airline carrier* field.

A more sophisticated authorization check is possible using the AUTHORITY-CHECK statement.

Within the transaction tz80 you can display and change flight data only if you have the appropriate authorization for the S\_CARRID authorization object in your user master record.

```

*&-----*
*&      Module  USER_COMMAND_0100  INPUT
*&-----*
MODULE USER_COMMAND_0100 INPUT.
  CASE OK_CODE.
    WHEN 'SHOW'.
      AUTHORITY-CHECK OBJECT 'S_CARRID'
        ID 'CARRID' FIELD '*'
        ID 'ACTVT'  FIELD '03'.

      IF SY-SUBRC NE 0. MESSAGE E009. ENDIF.
      MODE = CON_SHOW.
      SELECT SINGLE * FROM SPFLI
        WHERE CARRID = SPFLI-CARRID
          AND  CONNID = SPFLI-CONNID.
      IF SY-SUBRC NE 0.
        MESSAGE E005 WITH SPFLI-CARRID SPFLI-CONNID.
      ENDIF.
      CLEAR OK_CODE.
      SET SCREEN 200.
    WHEN 'CHNG'.
      AUTHORITY-CHECK OBJECT 'S_CARRID'
        ID 'CARRID' FIELD '*'
        ID 'ACTVT'  FIELD '02'.
      IF SY-SUBRC NE 0. MESSAGE E010. ENDIF.
      MODE = CON_CHANGE.
      SELECT SINGLE * FROM SPFLI
        WHERE CARRID = SPFLI-CARRID
          AND  CONNID = SPFLI-CONNID.
      IF SY-SUBRC NE 0.

```

## Defining an Authority Check

```

        MESSAGE E005 WITH SPFLI-CARRID SPFLI-CONNID.
    ENDIF.
    OLD_SPFLI = SPFLI.
    CLEAR OK_CODE.
    SET SCREEN 200.
ENDCASE.
ENDMODULE.                " USER_COMMAND_0100  INPUT

```

If you select the *Display* function in the transaction, the **AUTHORITY-CHECK** statement checks for the value '03' in the ACTVT field in the S\_CARRID authorization object. If you select the *Change* function, then the value '02' has to be present in the ACTVT field.

You can find more information about setting up authorizations for user master records in *BC Users and Authorizations*.

---

## Defining Authorization Objects

If there is no suitable authorization object for a transaction that needs protecting, you can set up an authorization object and fields yourself.

Authorization objects can be defined from the *ABAP/4 Development Workbench* menu by selecting *Development* → *Other tools* → *Authorization objs* → *Objects*.

Each authorization object must be assigned to an object class. These classes are organized according to the components of the system. You can assign a new object to the object class of the component with which you are working or create a new class. If you do so, select class names that begin with Y or Z to avoid conflicts with SAP names.

To define a new authorization object, enter a unique object name and the fields that belong to the object.



### Note

The second sign of the object name shouldn't be an underscore. The underscore is used by SAP objects and might be overwritten when new releases are installed.

---

## Defining Authorization Fields

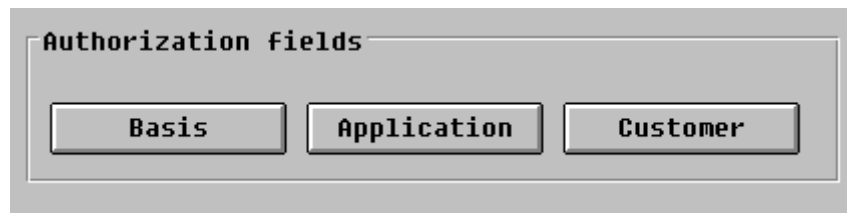
Each object consists of up to ten fields, which the programmer can define. Programmers can define authorization fields in the *ABAP/4 Development Workbench* menu by selecting *Development* → *Other tools* → *Authorization objs* → *Fields*.



### Note

Before you define a field, check that you have defined the data dictionary structure ZAUTHCUST. This structure is required for customer defined authorization fields. For further information on defining this structure, see *BC Users and Authorizations*.

To define new authorization fields, enter the name of the field and associate an ABAP/4 dictionary data element with it. To avoid having your fields deleted when you install a new release, define your fields only with the customer function in field maintenance. The customer function automatically saves your fields in the ZAUTHCUST structure.



You can often reuse fields defined by SAP in your own authorization objects. It is not required that you define your own fields if you define a new authorization object. For example you can use the SAP field ACTVT in your own authorization objects to represent a wide variety of actions in the system.