Objectives

In this Chapter we will discuss

- **Overview of LDB**
- **Usage of LDB in ABAP Program**
- **Advantages of LDB**

- Two ways of accessing data from database tables
  - **Accessing data using SELECT.**

    You can read and analyze data from all database tables known to the SAP system by using SELECT statement with its different clauses.

  - **Accessing data using Logical Database(LDB).**

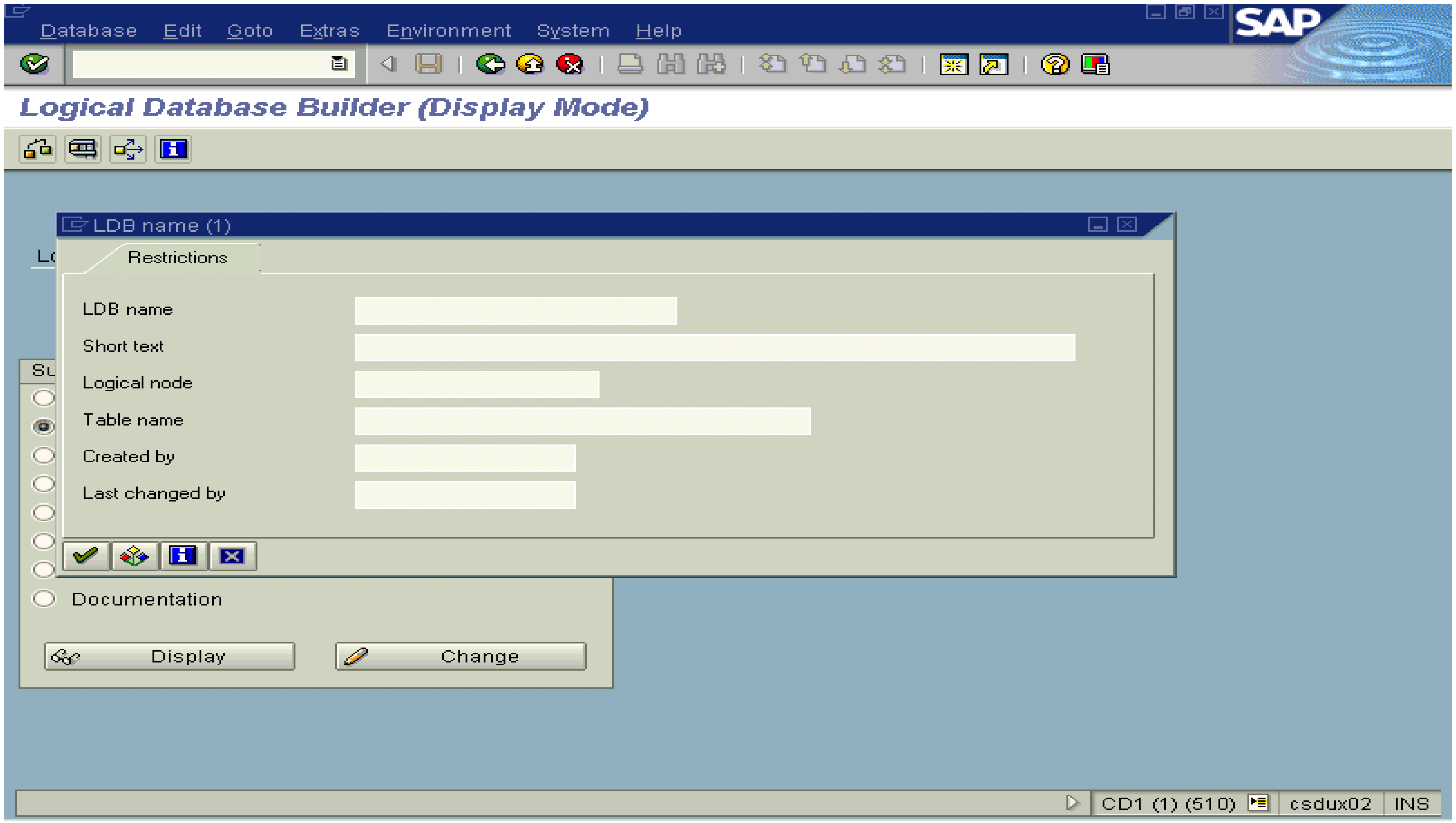
    Logical databases provide a method for accessing data in database tables which differs from the SELECT statement. You can link a logical database to an ABAP/4 report program as an attribute , it then supplies the report program with a set of hierarchically structured table lines which can be taken from different database tables.

Report with SELECT statements	Report with LDB
Report ……..	Report ……..
Tables spfli, sflight, sbook.	Tables spfli, sflight, sbook.
SELECT * from spfli where ...	Get spfli.
<processing block>	<processing block>
SELECT * from sflight where …	Get sflight.
<processing block>	<processing block>
SELECT * from sbook where …	Get sbook.
<processing block>	<processing block>
ENDSELECT.	
ENDSELECT.	
ENDSELECT.	

- **An easy-to-use standard user interface**
- **Meaningful data selection**
- **Central authorization checks for database accesses**
- **Check functions which check that user input is complete , correct and plausible**
- **Good read access performance while retaining the hierarchical data view determined by the application logic**
- **Create and design your own selection screen versions**

Go to transaction 'SE36' .Click F4 for list of LDBs' . You will get the following screen .

Fill up the relevant fields and pick up the required LDB .


Database Edit Goto Extras Environment System Help
SAP
Logical Database Builder (Display Mode)
LDB name (1)
Restrictions
Documentation
Display
Change
CD1 (1) (510)
csdux02
INS

LDB name	
Short text	
Logical node	
Table name	
Created by	
Last changed by	

Logical databases allow you to program several different tasks centrally . You use the Logical database to perform the following tasks :

– If several reports read the same data , you can code the read accesses in a single LDB.

– If you want to use the same user interface for several reports , you can implement this easily with the selection screens of logical database .

– Authorization checks for central data are coded centrally in logical database .

– If you want to improve response times , logical databases permit you to take a number of measures to achieve this .

- Structure
- Selections
- Database program

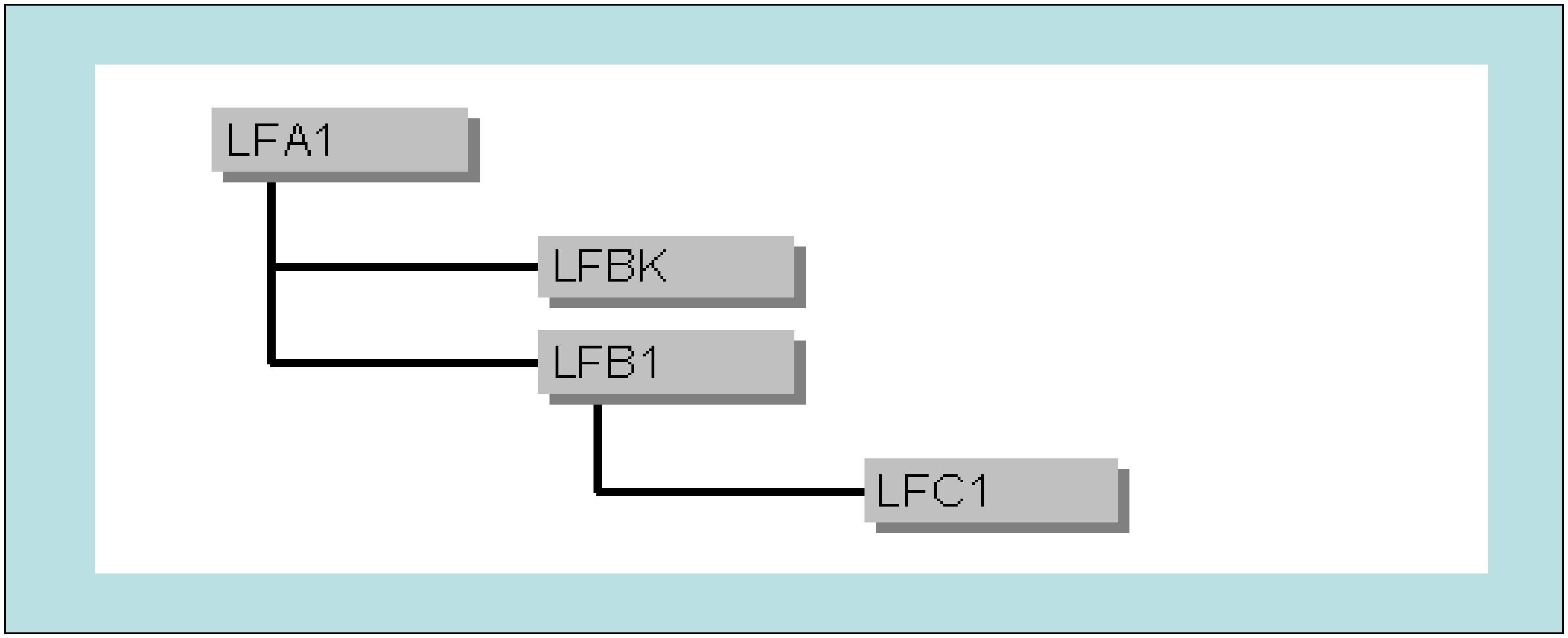
- Structure of LDB

  The structure of a logical database is usually based on the foreign key relationships between hierarchical tables in the R/3 System.

  Logical databases have a tree-like structure, which can be defined as follows:

  - **There is a single node at the highest level. This is known as the root node.**
  - **Each node can have one or several branches.**
  - **Each node is derived from one other node.**

- Structure of LDB



- **Selections**

  The selections in a logical database are defined using the normal statements for defining selection screens, that is,

  PARAMETERS

  SELECT-OPTIONS

  SELECTION-SCREEN

  – **Example**

```
SELECT-OPTIONS SLIFNR FOR LFA1-LIFNR.
PARAMETERS PBUKRS LIKE LFB1-BUKRS FOR
                    TABLE LFB1.
```

- **Selections**

  - **Dynamic Selections**

    The tables defined in the structure can have dynamic selections using the following code in the Selections Include program

    SELECTION-SCREEN DYNAMIC SELECTION FOR TABLE <tbnam>

  - **Field Selections**

    The tables for which field selection is defined can be called from the ABAP/4 program using GET <tbnam> Fields <f1> ….<fn> addition . The code for field selection in the Selections Include program is :

    SELECTION-SCREEN FIELD SELECTION FOR TABLE <tbnam>

- Database Program

  The name of the database program of a logical database <ldb> conforms to the naming convention SAPDB<ldb>.

  It serves as a container for subroutines, which the ABAP runtime environment calls when a logical database is processed.

- **Database Program**
  - – **Subroutines in LDB**

FORM LDB_PROCESS_INIT

**Called once only before the logical database is processed.
It prepares it to be called more than once by the function module
LDB_PROCESS.**

FORM INIT

Called once only before the selection screen is processed.

FORM PBO

**Called before the selection screen is displayed,
each time it is displayed. Consequently, it is only called when you
use the logical database with an executable program,
not with the function module LDB_PROCESS**

- **Database Program**
  - **Subroutines in LDB**

FORM PAI

Called when the user interacts with the selection screen. Consequently, it is only called when you use the logical database with an executable program, not with the function module LDB_PROCESS.

FORM LDB_PROCESS_CHECK_SELECTIONS

Called instead of the subroutine PAI if the logical database is called using the function module LDB_PROCESS without a selection screen.
This subroutine can check the selections passed in the function module interface.

- Database Program
  - Subroutines in LDB

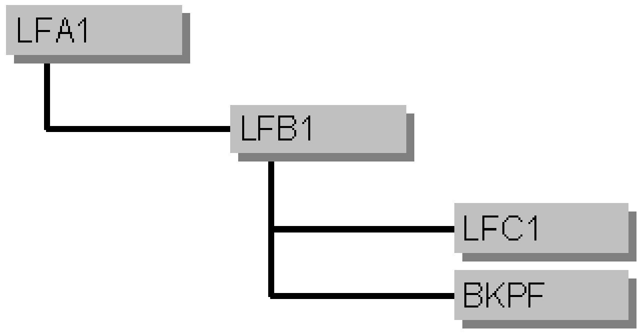
    FORM PUT_<node>

    Called in the sequence defined in the structure.
    Reads the data from the node <node> and uses the

    PUT <node>.

    Statement to trigger a corresponding GET event in the ABAP runtime environment.

- **Structure**



- **Selections in the Selection Include**

```
SELECT-OPTIONS:   SLIFNR   FOR LFA1-LIFNR,
                  SBUKRS   FOR LFB1-BUKRS,
                  SGJAHR   FOR LFC1-GJAHR,
                  SBELNR   FOR BKPF-BELNR.
```

```
*----------------------------------------------------*
* DATABASE PROGRAM OF THE LOGICAL
* DATABASE TEST_LDB                              *
*----------------------------------------------------*
PROGRAM SAPDBTEST_LDB DEFINING
DATABASE TEST_LDB.

TABLES:  LFA1,
          LFB1,
          LFC1,
          BKPF.
*----------------------------------------------------*
* Initialize selection screen (process before PBO)
*----------------------------------------------------*
FORM INIT.
    ....
ENDFORM.                          "INIT
*----------------------------------------------------*
* PBO of selection screen (always before selection
* screen
*----------------------------------------------------*
FORM PBO.
   ....
ENDFORM.                          "PBO
```

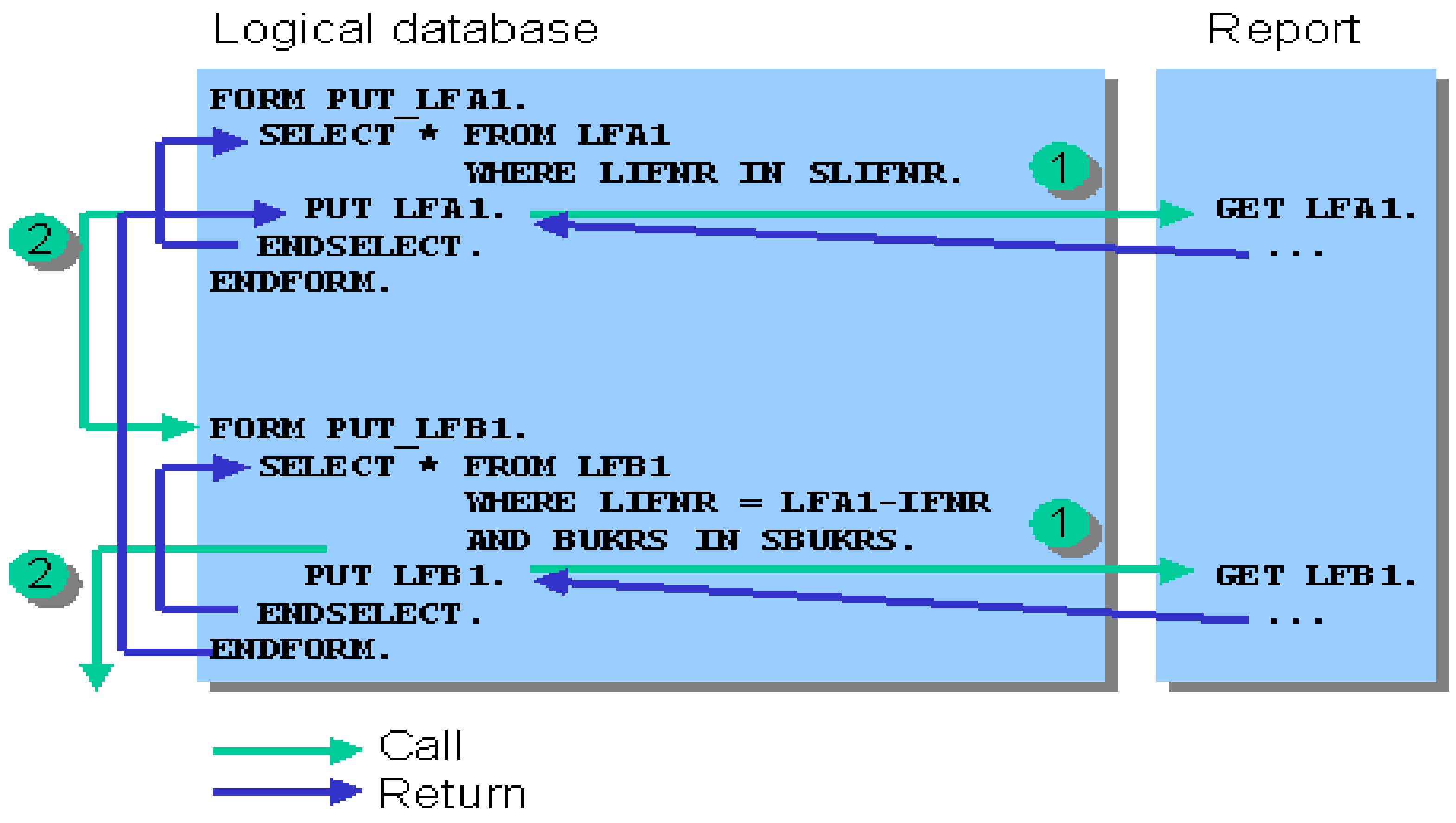
```
*---------------------------------------------------------*
* PAI of selection screen (process always after ENTER)
*---------------------------------------------------------*
FORM PAI USING FNAME MARK.
 CASE FNAME.
    WHEN 'SLIFNR'.
         ....
    WHEN 'SBUKRS'.
         ....
    WHEN 'SGJAHR'.
         ....
    WHEN 'SBELNR'.
         ....
  ENDCASE.
ENDFORM.                              "PAI
*---------------------------------------------------------*
* Call event GET LFA1
*---------------------------------------------------------*
```

```
 FORM PUT_LFA1.
           SELECT * FROM LFA1
                     WHERE LIFNR IN SLIFNR.
             PUT LFA1.
           ENDSELECT.
ENDFORM.                    "PUT_LFA1

*----------------------------------------------------*
* Call event GET LFB1
*----------------------------------------------------*
FORM PUT_LFB1.
           SELECT * FROM LFB1
                   WHERE LIFNR = LFA1-LIFNR

                           AND BUKRS IN SBULRS.
           CASE FNAME.
                     WHEN 'SLIFNR'.
                                ....
                     WHEN 'SBUKRS'.
                                ....
                     WHEN 'SGJAHR'.
                                ....
                     WHEN 'SBELNR'.
                                ....
           ENDCASE.
ENDFORM.                    "PAI
```

```
*-----------------------------------------------------*
* Call event GET LFA1
*-----------------------------------------------------*

PUT LFB1.
             ENDSELECT.
ENDFORM.                          "PUT_LFB1


*-----------------------------------------------------*
* Call event GET LFC1
*-----------------------------------------------------*

FORM PUT_LFC1.
             SELECT * FROM LFC1
                          WHERE LIFNR = LFA1-LIFNR
                          AND BUKRS = LFB1-BUKRS
                          AND GJAHR IN SGJAHR.
                    PUT LFC1.
              ENDSELECT.
```

```
ENDFORM.                    "PUT_LFC1

*-------------------------------------------------------*
* Call event GET BKPF
*-------------------------------------------------------*

FORM PUT_BKPF.
        SELECT * FROM BKPF
                WHERE BUKRS = LFB1-BUKRS
                AND BELNR IN SBELNR
                AND GJAHR IN SGJAHR.
          PUT BKPF.
        ENDSELECT.

ENDFORM.                    "PUT_BKPF
```

- **Relation between PUT and GET statements**

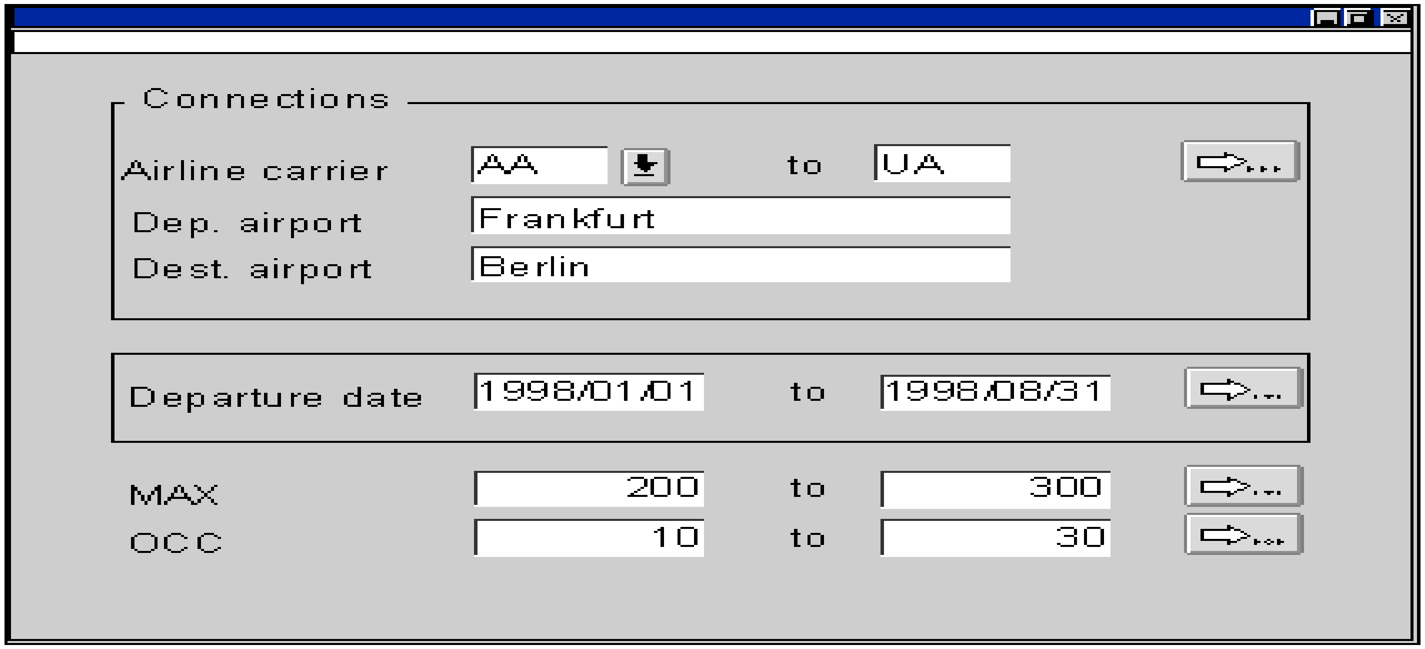


```
FORM PUT_LFA1.
  SELECT * FROM LFA1
           WHERE LIFNR IN SLIFNR.
    PUT LFA1.
  ENDSELECT.
ENDFORM.

FORM PUT_LFB1.
  SELECT * FROM LFB1
           WHERE LIFNR = LFA1-IFNR
           AND BUKRS IN SBUKRS.
    PUT LFB1.
  ENDSELECT.
ENDFORM.
```

```
GET LFA1.
...

GET LFB1.
...
```

- **Relation between PUT and GET statements**

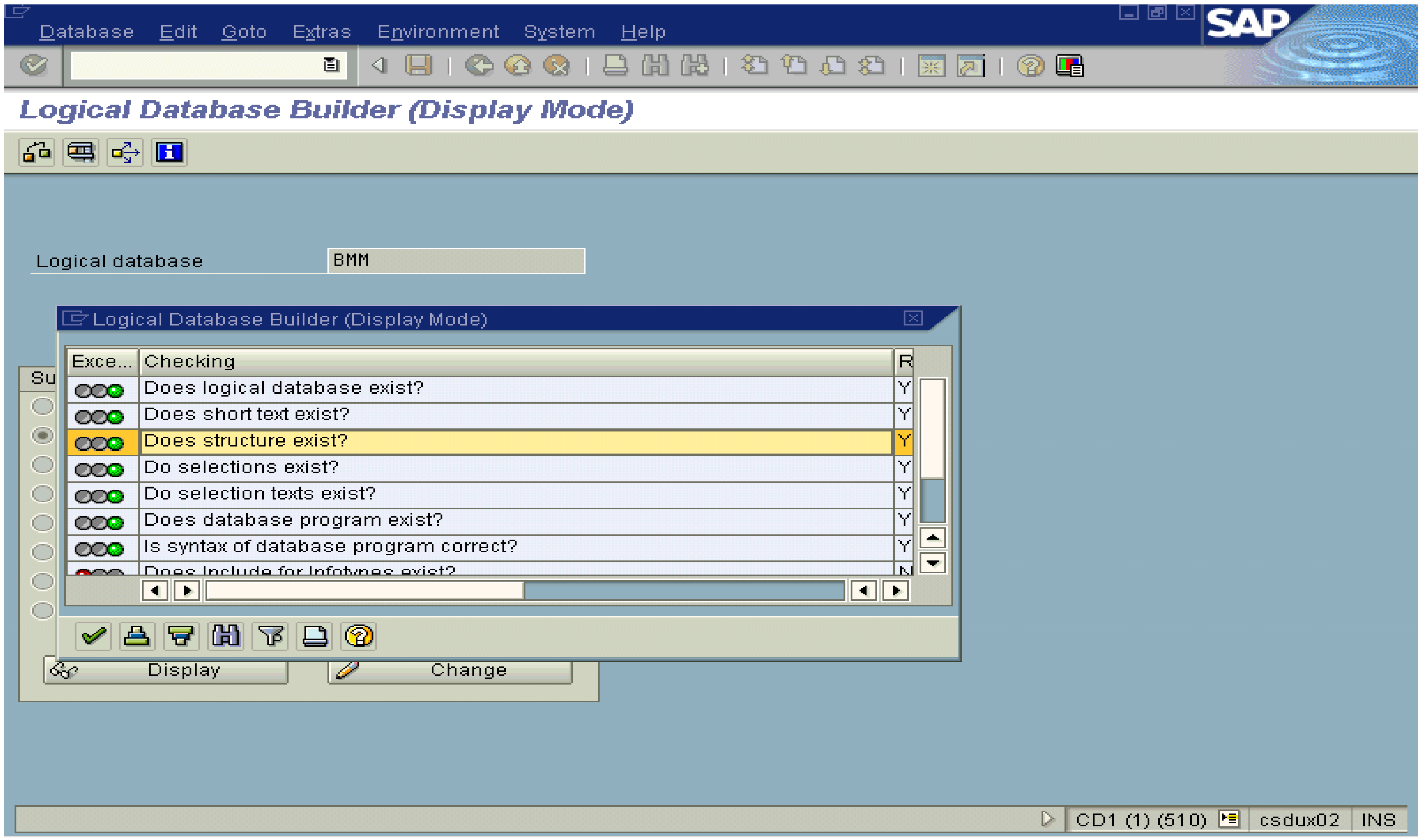
```
REPORT DEMO.
NODES: SPFLI,SFLIGHT.
GET SFLIGHT.
WRITE: / SPFLI-CARRID, SPFLI-CONNID.
```



Connections

Airline carrier	AA	to	UA
Dep. airport	Frankfurt		
Dest. airport	Berlin		
Departure date	1998/01/01	to	1998/08/31
MAX	200	to	300
OCC	10	to	30

- To check whether a logical database is correct and complete , choose Check on the initial screen.Then see a screen which displays these checks:



Logical Database Builder (Display Mode)

Logical database BMM

Logical Database Builder (Display Mode)

Exce...	Checking	R
●●●	Does logical database exist?	Y
●●●	Does short text exist?	Y
●●●	Does structure exist?	Y
●●●	Do selections exist?	Y
●●●	Do selection texts exist?	Y
●●●	Does database program exist?	Y
●●●	Is syntax of database program correct?	Y
●●●	Does Include for Infotypes exist?	N

CD1 (1) (510) csdux02 INS

•Two methods for database selections

•Comparison between the two methods

•Components of LDB

•Task of LDB

•Advantages of LDB

•Linking LDB in ABAP Report

•Searching LDB

•Example of LDB

•Checking LDB

- Exercise 1
  - **Write a report for getting the Purchasing document number using LDB ‘EMM’**

```
REPORT Z_LOG_DATA_EXP1 .

TABLES: EKKO.

*Get the data from the LDB
GET EKKO.

IF SY-SUBRC EQ 0.
*Write purchasing document number
  WRITE: / EKKO-EBELN.
ENDIF.
```

- Exercise 2
  - **Write a report for getting the Purchasing document number using LDB 'EMM' using the GET statement with FIELDS addition**

```
REPORT Z_LOG_DATA_EXP1 .

TABLES: EKKO.

*Get only the purchasing document number  from the LDB
GET EKKO FIELDS EBELN.

IF SY-SUBRC EQ 0.

*Write the purchasing document number
  WRITE: / EKKO-EBELN.

ENDIF.
```

- Exercise 3
  - **Write a report for getting the Purchasing document number using LDB ‘EMM’ using the GET LATE statement**

```
REPORT Z_LOG_DATA_EXP3 .

TABLES: EKKO, EKPO, EKET.

* This GET statement will be executed after all GET statements
GET EKKO LATE FIELDS EBELN .

IF SY-SUBRC EQ 0.

*Write the purchasing document number
  WRITE: / ’Purchasing Document number ', EKKO-EBELN.

ENDIF.
```

- Exercise 3
  - **Write a report for getting the Purchasing document number using LDB ‘EMM’ using the GET statement with FIELDS addition**

```
GET EKPO.

IF SY-SUBRC EQ 0.
* Write the material number
  WRITE: / ’Material Number', EKPO-MATNR.
ENDIF.

GET EKET.

IF SY-SUBRC EQ 0.

* Write the quantity
  WRITE: / ’Quantity', EKET-WEMNG.

ENDIF
```

- Exercise 4
  - **Write a report for hiding the fields on the selection screen generated by the LDB 'EMM'**

```
REPORT Z_LOG_DATA_EXP4 .

TABLES: EKKO, EKPO.

AT SELECTION-SCREEN OUTPUT.

* Loop at  screen internal table
LOOP AT SCREEN .

  IF SCREEN-NAME = '%_EM_WERKS_%_APP_%-TEXT'.
  SCREEN-INVISIBLE = '1' .
  MODIFY SCREEN.
  ENDIF.
```

- Exercise 4
  - **Write a report for hiding the fields on the selection screen generated by the LDB ‘EMM’**

```
IF SCREEN-NAME = 'EM_WERKS-LOW'.
   SCREEN-INVISIBLE = '1' .
   SCREEN-INPUT = '0'.
    MODIFY SCREEN.
  ENDIF.

 IF SCREEN-NAME = 'EM_WERKS-HIGH'.
  SCREEN-INVISIBLE = '1' .
   SCREEN-INPUT = '0'.
   MODIFY SCREEN.
  ENDIF.
```

- Exercise 4
  - **Write a report for hiding the fields on the selection screen generated by the LDB ‘EMM’**

```
IF SCREEN-NAME = '%_EM_WERKS_%_APP_%-VALU_PUSH'.
  SCREEN-INVISIBLE = '1' .
  MODIFY SCREEN.
ENDIF.


ENDLOOP.


START-OF-SELECTION.

GET EKKO LATE .

IF SY-SUBRC EQ 0 .
```

- Exercise 4
  - **Write a report for hiding the fields on the selection screen generated by the LDB ‘EMM’**

```
WRITE: / ’Purchasing Document Number ', EKKO-EBELN.

ENDIF.

GET EKPO FIELDS MATNR.

IF SY-SUBRC EQ 0.

WRITE: / ’Material Number ', EKPO-MATNR.

ENDIF.
```