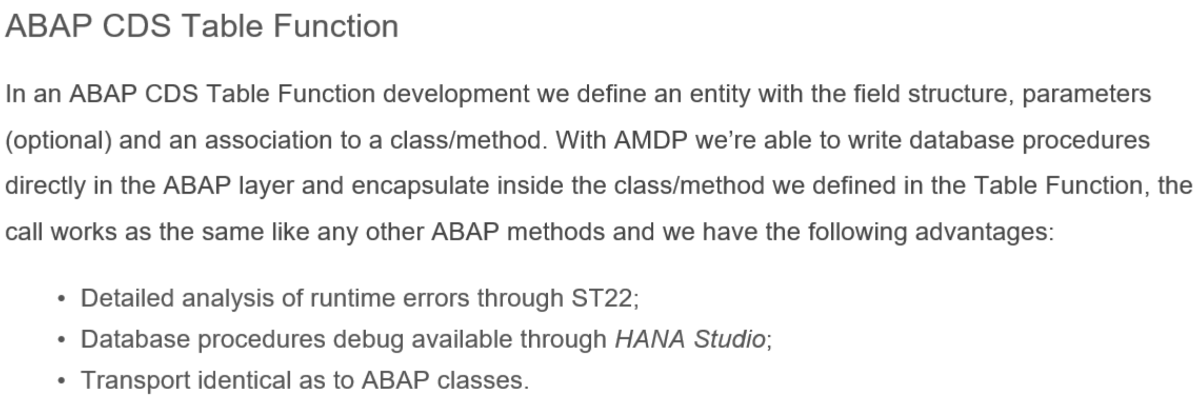
CDS

table function with

Select option

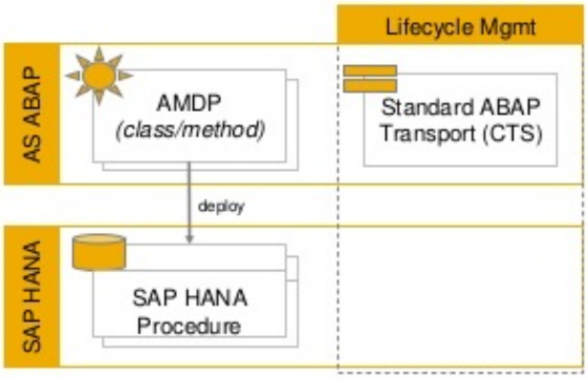
When it isn’t possible to solve your scenario with an ABAP CDS View there is an alternative solution creating an ABAP CDS Table Function powered by ABAP Managed Database Procedures (AMDP).



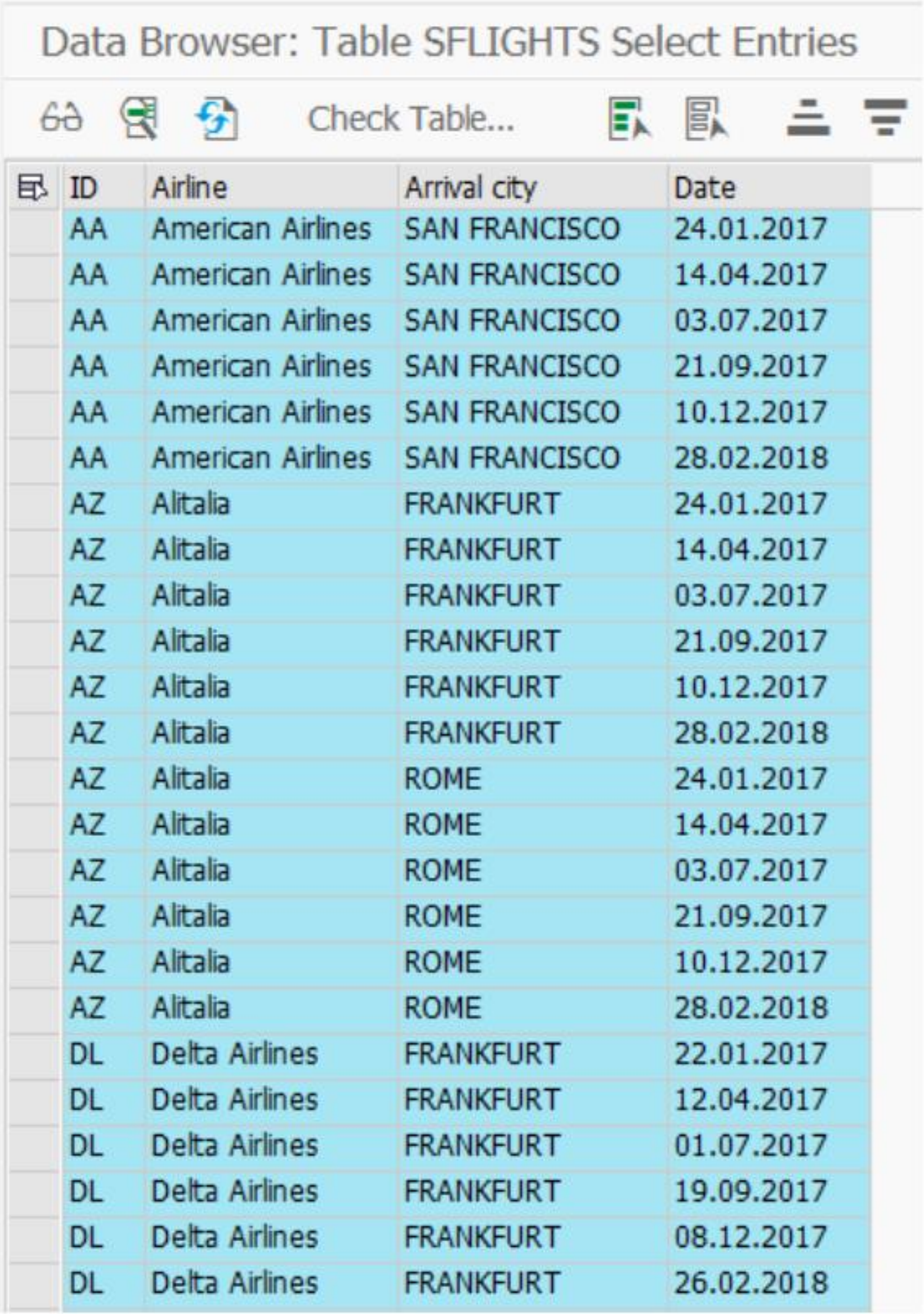
ABAP CDS Table Function

In an ABAP CDS Table Function development we define an entity with the field structure, parameters (optional) and an association to a class/method. With AMDP we’re able to write database procedures directly in the ABAP layer and encapsulate inside the class/method we defined in the Table Function, the call works as the same like any other ABAP methods and we have the following advantages:

- Detailed analysis of runtime errors through ST22;
- Database procedures debug available through *HANA Studio*;
- Transport identical as to ABAP classes.



Example to demonstrate Use of CDS Table Function:



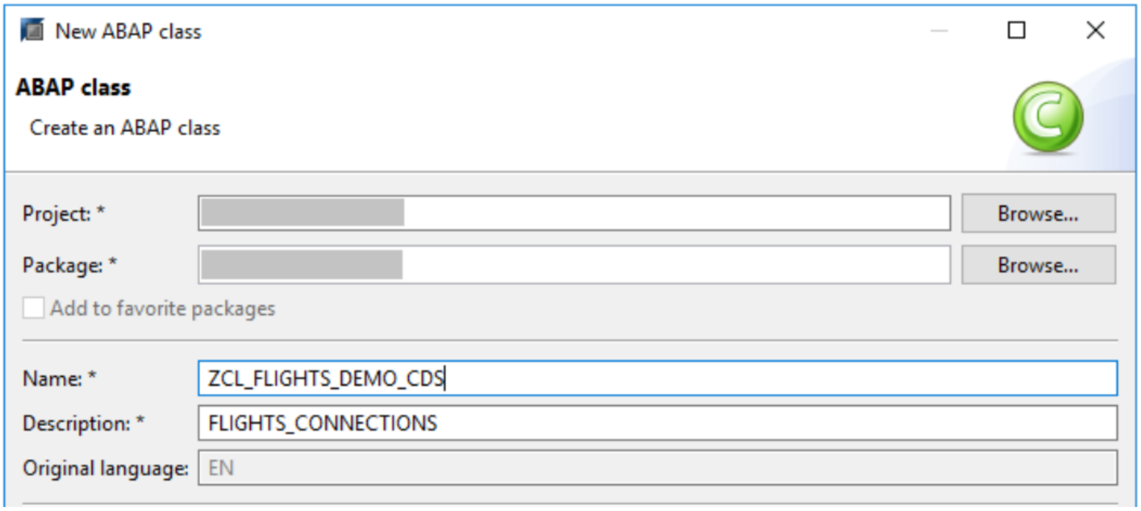
Data Browser: Table SFLIGHTS Select Entries

ID	Airline	Arrival city	Date
AA	American Airlines	SAN FRANCISCO	24.01.2017
AA	American Airlines	SAN FRANCISCO	14.04.2017
AA	American Airlines	SAN FRANCISCO	03.07.2017
AA	American Airlines	SAN FRANCISCO	21.09.2017
AA	American Airlines	SAN FRANCISCO	10.12.2017
AA	American Airlines	SAN FRANCISCO	28.02.2018
AZ	Alitalia	FRANKFURT	24.01.2017
AZ	Alitalia	FRANKFURT	14.04.2017
AZ	Alitalia	FRANKFURT	03.07.2017
AZ	Alitalia	FRANKFURT	21.09.2017
AZ	Alitalia	FRANKFURT	10.12.2017
AZ	Alitalia	FRANKFURT	28.02.2018
AZ	Alitalia	ROME	24.01.2017
AZ	Alitalia	ROME	14.04.2017
AZ	Alitalia	ROME	03.07.2017
AZ	Alitalia	ROME	21.09.2017
AZ	Alitalia	ROME	10.12.2017
AZ	Alitalia	ROME	28.02.2018
DL	Delta Airlines	FRANKFURT	22.01.2017
DL	Delta Airlines	FRANKFURT	12.04.2017
DL	Delta Airlines	FRANKFURT	01.07.2017
DL	Delta Airlines	FRANKFURT	19.09.2017
DL	Delta Airlines	FRANKFURT	08.12.2017
DL	Delta Airlines	FRANKFURT	26.02.2018

Requirement:
Concatenate multiple records in a single field using ABAP CDS Table Function:

```
define table function ZDEMO_FLIGHTS_TABLE_FUNCTION
returns
{
  client       : abap.clnt;
  airline_code : s_carr_id;
  airline_name : s_carrname;
  cities_to    : abap.string;
}
implemented by method
  ZCL_FLIGHTS_DEMO_CDS=>FLIGHTS_CONNECTIONS;
```

Let’s create a new ABAP class with the name **ZCL_FLIGHTS_DEMO_CDS**, select a transport and click in *Finish*.



New ABAP class

ABAP class

Create an ABAP class

Project: * [redacted]

Package: * [redacted]

□ Add to favorite packages

Name: * ZCL_FLIGHTS_DEMO_CDS

Description: * FLIGHTS_CONNECTIONS

Original language: EN

Adapt your class including the interface **IF_AMDP_MARKER_HDB**. This step will transform the ABAP class into an AMDP class and provide the possibility to include database procedures inside its methods.

```
PUBLIC SECTION.
  INTERFACES if_amdp_marker_hdb.
```

Declare a new public method and include the statement **FOR TABLE FUNCTION** referencing the table function we created in the first step.

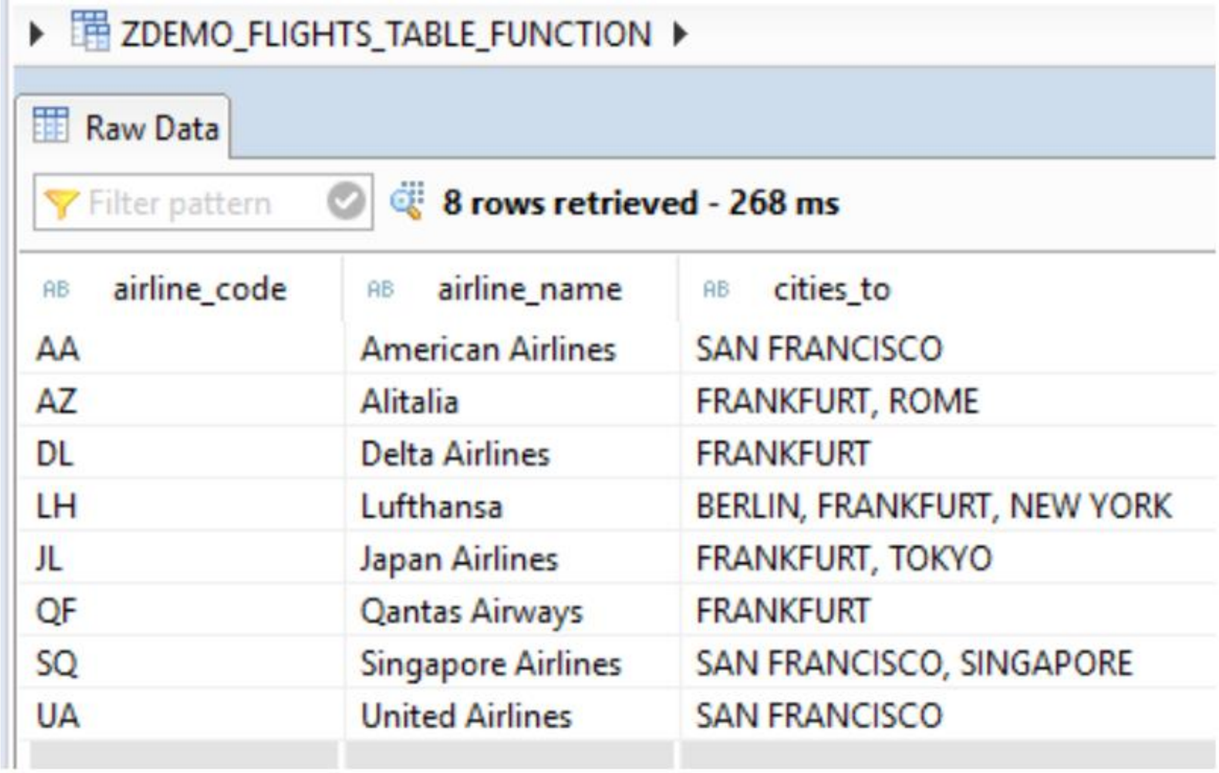
```
CLASS-METHODS:
  flights_connections FOR TABLE FUNCTION ZDEMO_FLIGHTS_TABLE_FUNCTION.
```

In the method implementation we need to include some configuration options:

- **BY DATABASE FUNCTION**: This option will mark the method as a table function, another option is to generate a procedure changing the statement to **BY DATABASE PROCEDURE**.
- **FOR HDB**: Defines the database type as HDB (HANA Database).
- **LANGUAGE SQLSCRIPT**: Language used by HANA database procedures.
- **OPTIONS READ-ONLY**: No changes allowed inside the database procedure.
- **USING**: Definition of database tables, views or procedures that would be consumed inside our table function. In our case we need to access data only from ***SFLIGHTS*** view.

```
CLASS zcl_flights_demo_cds DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_amdp_marker_hdb.

    CLASS-METHODS:
      flights_connections FOR TABLE FUNCTION zdemo_flights_table_function.

  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_flights_demo_cds IMPLEMENTATION.

  METHOD flights_connections
    BY DATABASE FUNCTION
    FOR HDB
    LANGUAGE SQLSCRIPT
    OPTIONS READ-ONLY
    USING sflights.
```

```
    itab_cities =
      SELECT DISTINCT
             sflights.mandt    as client,
             sflights.carrid   as airline_code,
             sflights.carrname as airline_name,
             sflights.cityto   as city_to
        FROM sflights;

    RETURN
      SELECT client,
             airline_code,
             airline_name,
             STRING_AGG(city_to, ', ' ORDER BY city_to) as cities_to
        FROM :itab_cities
       GROUP BY client,
                airline_code,
                airline_name;

  ENDMETHOD.

ENDCLASS.
```



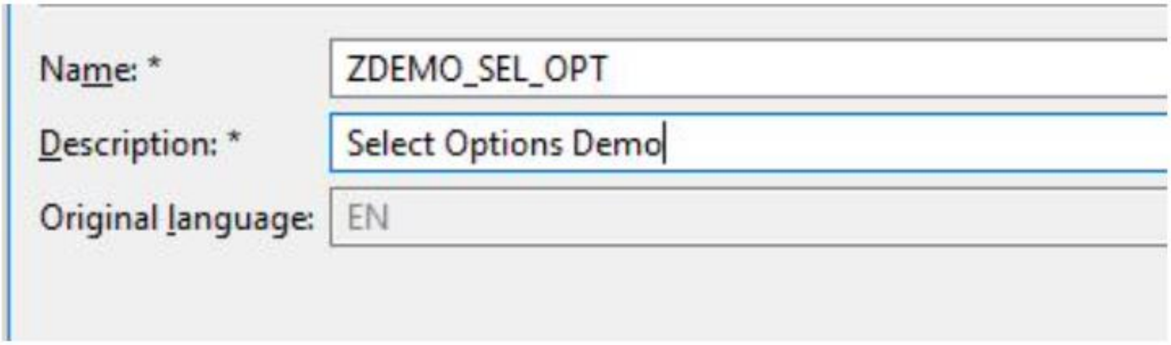
ZDEMO_FLIGHTS_TABLE_FUNCTION

8 rows retrieved - 268 ms

airline_code	airline_name	cities_to
AA	American Airlines	SAN FRANCISCO
AZ	Alitalia	FRANKFURT, ROME
DL	Delta Airlines	FRANKFURT
LH	Lufthansa	BERLIN, FRANKFURT, NEW YORK
JL	Japan Airlines	FRANKFURT, TOKYO
QF	Qantas Airways	FRANKFURT
SQ	Singapore Airlines	SAN FRANCISCO, SINGAPORE
UA	United Airlines	SAN FRANCISCO

Requirement:

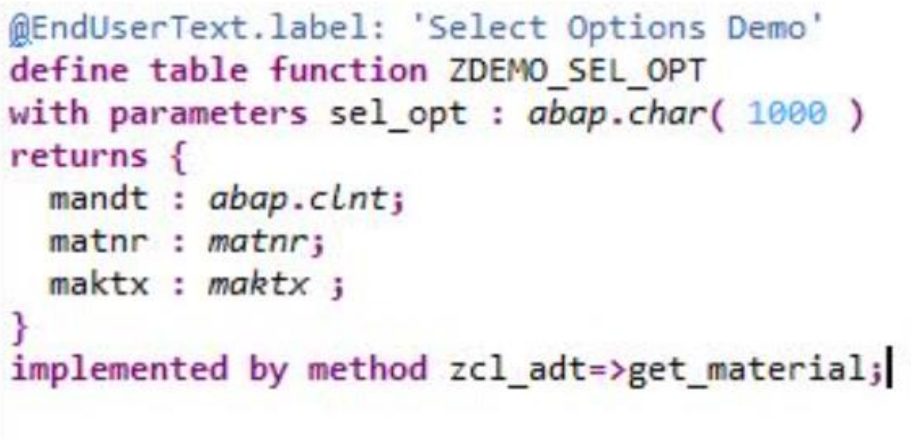
2) Select-Options in CDS View which it will return Material no & Description.



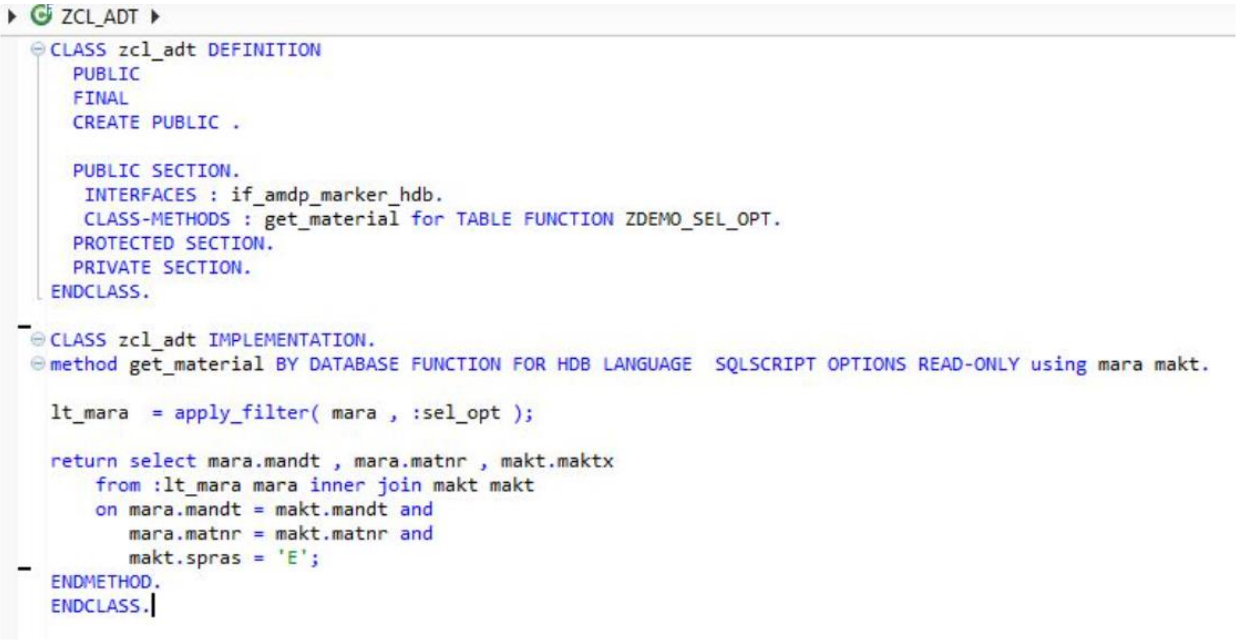
Name: * ZDEMO_SEL_OPT

Description: * Select Options Demo

Original language: EN



```
@EndUserText.label: 'Select Options Demo'
define table function ZDEMO_SEL_OPT
with parameters sel_opt : abap.char( 1000 )
returns {
  mandt : abap.clnt;
  matnr : matnr;
  maktx : maktx ;
}
implemented by method zcl_adt=>get_material;
```



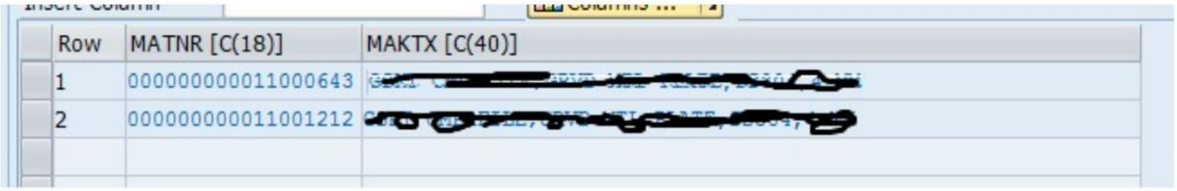
```
CLASS zcl_adt DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
   INTERFACES : if_amdp_marker_hdb.
   CLASS-METHODS : get_material for TABLE FUNCTION ZDEMO_SEL_OPT.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_adt IMPLEMENTATION.
method get_material BY DATABASE FUNCTION FOR HDB LANGUAGE  SQLSCRIPT OPTIONS READ-ONLY using mara makt.

lt_mara  = apply_filter( mara , :sel_opt );

return select mara.mandt , mara.matnr , makt.maktx
    from :lt_mara mara inner join makt makt
    on mara.mandt = makt.mandt and
       mara.matnr = makt.matnr and
       makt.spras = 'E';
ENDMETHOD.
ENDCLASS.
```

Now the ABAP side.

```
data : l_matnr type matnr,
       lt_tab type TABLE OF zdemo_sel_opt,
       L_WHERE TYPE STRING.
SELECT-OPTIONS : s_matnr for l_matnr.


START-OF-SELECTION.

L_WHERE = CL_SHDB_SELTAB=>combine_seltabs(
                                      EXPORTING it_named_seltabs =
                                            value #( ( name = 'MATNR' dref = REF #( s_matnr[] ) )
                                            iv_client_field = 'MANDT'
                                ).
```

```
select * from zdemo_sel_opt( sel_opt = @l_where ) into table @lt_tab.
```

Output:



Row	MATNR [C(18)]	MAKTX [C(40)]
1	000000000011000643	[redacted]
2	000000000011001212	[redacted]