Products Industries Support Training Community Developer Partner

About

Ask a Question Write a Blog Post

Login

Carine Tchoutouo Djomo

January 15, 2014 8 minute read

Implement and consume your first ABAP Managed Database Procedure on HANA

Follow RSS feed

3 Likes 34,607 Views 25 Comments

This tutorial demonstrates how to use the new the “Top-Down”-approach for consuming HANA procedures provided by means of ABAP Managed Database Procedures delivered with SAP NetWeaver AS ABAP 7.4, Support Package 5.

You’ll be guided in a step-by-step manner over the creation, implementation and consumption of an ABAP Managed Database Procedure (AMDP) on HANA.

For a quick and comprehensive introduction into the ABAP Managed Database Procedures, read the Introduction into AMDP written by Jens Weiler and also check the ABAP Development Tools Help (menu entry `Help > Help Content`). Information, guides and tutorials about the development of ABAP for SAP HANA applications – meaning applications built out of ABAP and HANA development entities-, visit our SCN Page.

> Note that the purpose of this tutorial is not to introduce the SQLScript programming language. You will find

more infomation on that topic in the SAP HANA SQLScript Reference.

Prerequisites

- SAP NetWeaver AS ABAP 7.4 Support Package 5 (or higher) running on SAP HANA
- SAP HANA Appliance Software SPS 05 (or higher)
- SAP HANA DB SQLScript V2.0 (or higher)
- ABAP Development Tools for SAP NetWeaver (version 2.19)

Tutorial Objectives

After completing this tutorial, you will be able to:

- Declare an AMDP class
- Declare an AMDP method
- Implement an AMDP method
- Consume an AMDP method in ABAP

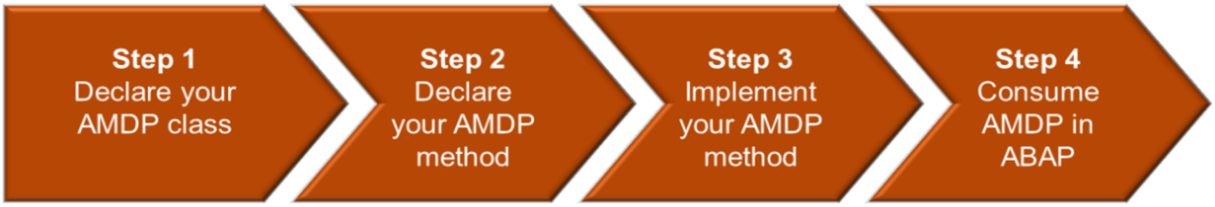
Use Case Description

The Account Receivables accountant of your company want to be able to display the so-called top and flop customers in regards to their payment ability based on the gross amount of the open invoices.

The company accountant should be able to select how many customers have to be displayed per category and due to the regular update of the business data, the categorization have to be executed on-the-fly. Millions of open item invoices are typically processed in such tasks.

More information about the Open Items Analysis reference scenario underlying this use case is available under http://scn.sap.com/docs/DOC-41248.

Procedure Overview



Step 1
Declare your AMDP class

Step 2
Declare your AMDP method

Step 3
Implement your AMDP method

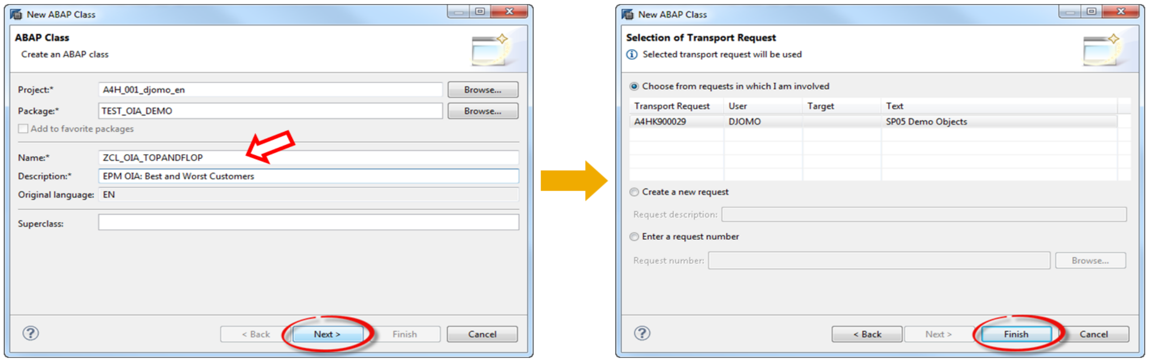
Step 4
Consume AMDP in ABAP

Step-by-Step Procedure

Step 1: Create an AMDP Class

In this step, you will create a regular global class and then mark it as AMDP class by specifying the tag interface for the SAP HANA database platform.

1. Start the ABAP Development Tools (aka ABAP in Eclipse) – or your SAP HANA Studio – and open the ABAP perspective by selecting menu entry `Window > Open perspective > Others…`, and choosing the `ABAP` entry in the appearing dialog box.
2. Go to your ABAP project in the `Project Explorer` and create a new class in the package of your choice by selecting the context menu entry `New… > ABAP Class`
3. Maintain the required information (e.g. `ZCL_OIA_TOPANDFLOP` as name and “`EPM OIA: Best and Worst Customers`” as description) and click on `Next`.
Select a transport request if required and confirm the creation dialog.



New ABAP Class

ABAP Class
Create an ABAP class

Project:* A4H_001_djomo_en
Package:* TEST_OIA_DEMO
□ Add to favorite packages
Name:* ZCL_OIA_TOPANDFLOP
Description:* EPM OIA: Best and Worst Customers
Original language: EN
Superclass:

New ABAP Class

Selection of Transport Request
Selected transport request will be used

● Choose from requests in which I am involved

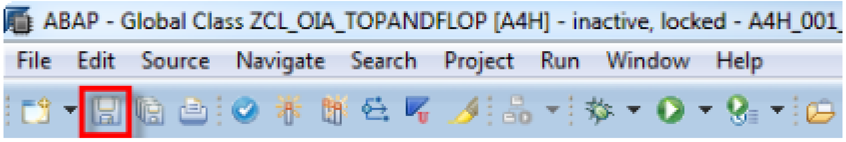
Transport Request	User	Target	Text
A4HK900029	DJOMO		SP05 Demo Objects

○ Create a new request
Request description:
○ Enter a request number
Request number:

4. Now go to the class definition and insert following line directly after `PUBLIC SECTION`:

```
    INTERFACESif_amdp_marker_hdb.
```

5. Save.



ABAP - Global Class ZCL_OIA_TOPANDFLOP [A4H] - inactive, locked - A4H_001_

You’ve just created your first AMDP class!

Step 2: Declare an AMDP Method

You will now declare the method `get_top_and_flop` which will be implemented as AMDP method later on. This method has an importing parameter for specifying the number of customers to be retrieved for each category (top and flop), and two exporting parameters getting for the two requested result sets.

> **Info**: An AMDP class can contain both regular ABAP methods and AMDP methods. An AMDP is declared like a regular static method or instance method in any visibility section. An AMDP method cannot be identified as such in the declaration part of the class. Nevertheless, the parameter interface of an AMDP method has to fulfill specific prerequisites. For example the parameters must be declared using `VALUE` for pass by value and return values cannot be declared using `RETURNING`.

Before going ahead with the method definition, we first have to defined the type of the result sets.

1. Define the two types `ty_bupa_selection` and `tt_bupa_selection` in the PUBLIC SECTION of the class definition – With `ty_bupa_selection` defining the table line of our return set, and `tt_bupa_selection` defining the type of the returned tables.
For that just copy the below coding after the interface declaration done in the previous steps:

```
TYPES:
  BEGIN OF ty_bupa_selection,
    company_name TYPE c LENGTH 80,
    gross_amount TYPE p LENGTH 8 DECIMALS 2,
  END OF ty_bupa_selection.

TYPES:
  tt_bupa_selection TYPE STANDARD TABLE OF ty_bupa_selection WITH EMPTY KEY.
```

2. Now define the public static method `get_top_and_flop`.
Just copy and pase the coding below directly after the type definitions in the public section:

```
CLASS-METHODS get_top_and_flop
  IMPORTING
    VALUE(iv_client) TYPE mandt
    VALUE(iv_number)  TYPE i
  EXPORTING
    VALUE(et_top)     TYPE tt_bupa_selection
    VALUE(et_flop)    TYPE tt_bupa_selection.
```

> **Info**: Analog to native SQL with ADBC and EXEC, the AMDP framework does not support automatic client handling. It means that in case of client-specific computations, the client context has to be passed to the AMDP method and used appropriately in the SQLScript coding.

3. An error will be displayed in the editor due to the missing method implementation. Just use the Quick Fix (`Ctrl+1`) function “`Add implementation for` **get_top_and_flop**” to quickly solved this issue.
4. Save your AMDP class.

Step 3: Implement the AMDP Method

You will now implement a relatively simple SQLScript-based AMDP method, which retrieves the best and worst customers depending on the gross amount of open invoices.

Info: Whether a method is implemented as ABAP or as AMDP method is not decided in the class definition, but rather in the class implementation.

An AMDP method is indicated as an AMDP method in the implementation part of the class using the addition `BY DATABASE PROCEDURE` of the statement `METHOD`. At the same time, the database platform where the method is used and the programming language used to implement the method are respectively specified with the additions `FOR` and `LANGUAGE`. Further additions are available.

1. Mark the method implementation as an AMDP method.
Go to the class definition and enhance the method with the required additions as displayed below:

```
METHOD get_top_and_flop BY DATABASE PROCEDURE
                          FOR HDB
                          LANGUAGE SQLSCRIPT
                          OPTIONS  READ-ONLY
USING snwd_so_i snwd_so snwd_bpa.

ENDMETHOD.
```

The compiler now knows, that the implementation of the method `get_top_and_flop` of the class `ZCL_OIA_TOPANDFLOP` is an SQLScript-based AMDP for the HANA database platform.The addition `USING` contains the name of the DDIC tables which will be used in the implementation.

2. Now implement the database procedure by copying the SQLScript source below

```
-retrieve the best customers
et_top = select top :iv_number bp.company_name as company_name,
sum(soi.gross_amount) as gross_amount
          from snwd_so_i as soi
          inner join snwd_so  as so
on so.node_key = soi.parent_key and so.client = soi.client
          inner join snwd_bpa as bp
            on bp.node_key = so.buyer_guid and bp.client = so.client
          group by company_name
          order by gross_amount desc;

-retrieve the worst customers
et_flop = select top :iv_number bp.company_name as company_name,
sum(soi.gross_amount) as gross_amount
          from snwd_so_i as soi
            inner join snwd_so  as so
on so.node_key = soi.parent_key and so.client = soi.client
            inner join snwd_bpa as bp
              on bp.node_key = so.buyer_guid and bp.client = so.client
            group by company_name
            order by gross_amount asc;
```

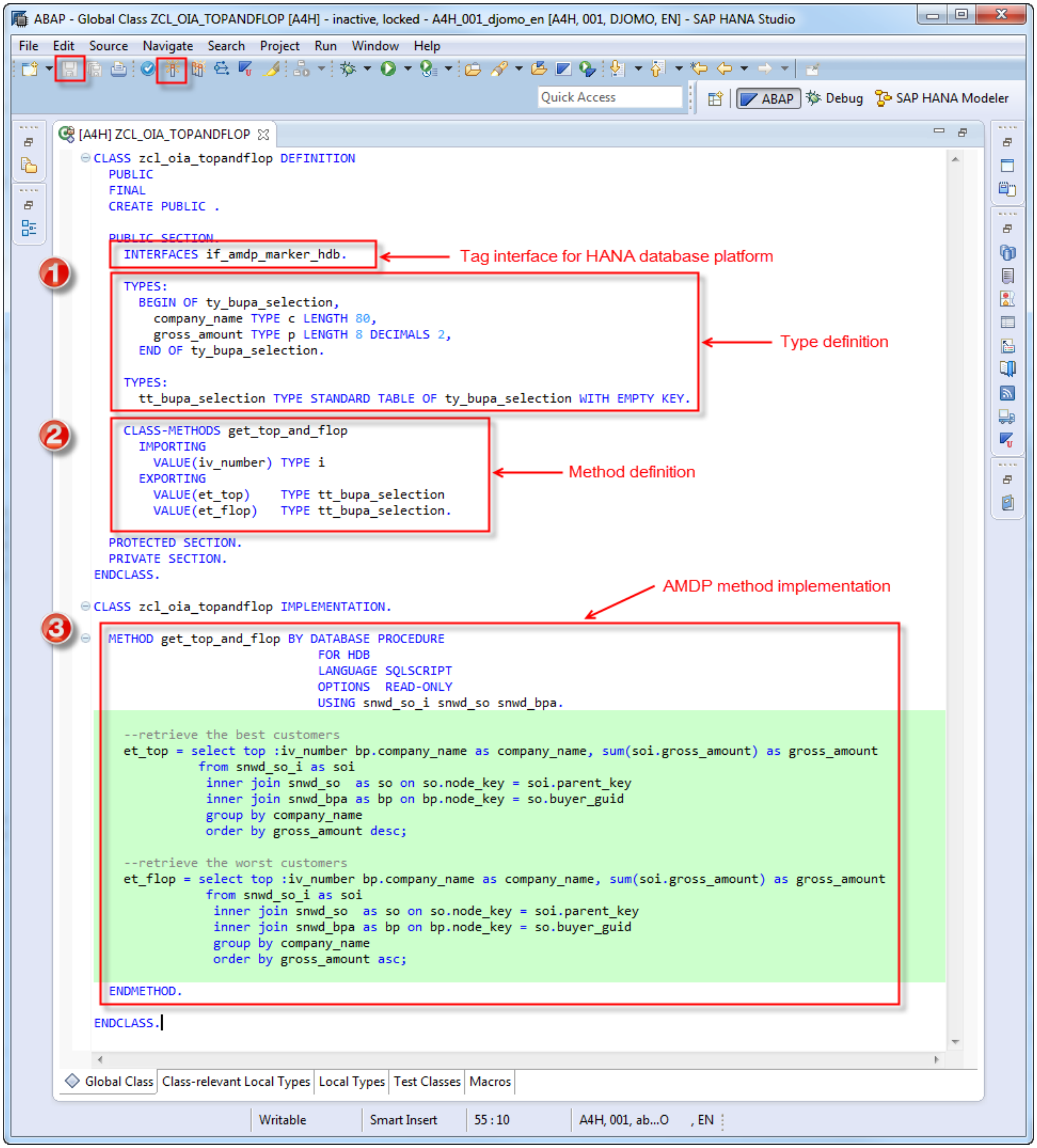
Note: The purpose of this tutorial is not to introduce the SQLScript programming language or to demonstrate how complex the logic of a database procedure can be. The above procedure implements a relatively simple data-intensive function, but of course very complex logic (making even use of the advanced HANA features and functions such as data mining and predictive analysis) can be implemented.

As already stated, in case of client-specific computation, the client context has to be passed to the AMDP method and used appropriately in the SQLScript coding. You may have a look at the SAP HANA SQLScript Reference.

Info: In order to quickly visualize whether a class contains AMDP methods and where, it is recommended to set a different background color for embedded languages – such as native SQL and SQLScript.
To achieve this, go to the ADT menu entry `Windows > Preferences` and select the path `General > Appearance > Color and Fonts > ABAP > Syntax Coloring > Embedded Languages (background color)` and set the background color of your choice.

3. Save and activate your AMDP class.

You’re now ready to test your AMDP method!


ABAP - Global Class ZCL_OIA_TOPANDFLOP [A4H] - inactive, locked - A4H_001_djomo_en [A4H, 001, DJOMO, EN] - SAP HANA Studio
File Edit Source Navigate Search Project Run Window Help
Quick Access
ABAP
Debug
SAP HANA Modeler
[A4H] ZCL_OIA_TOPANDFLOP
1
Tag interface for HANA database platform
Type definition
2
Method definition
AMDP method implementation
3
Global Class
Class-relevant Local Types
Local Types
Test Classes
Macros
Writable
Smart Insert
55 : 10
A4H, 001, ab...O
, EN

```
CLASS zcl_oia_topandflop DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_amdp_marker_hdb.

    TYPES:
      BEGIN OF ty_bupa_selection,
        company_name TYPE c LENGTH 80,
        gross_amount TYPE p LENGTH 8 DECIMALS 2,
      END OF ty_bupa_selection.

    TYPES:
      tt_bupa_selection TYPE STANDARD TABLE OF ty_bupa_selection WITH EMPTY KEY.

    CLASS-METHODS get_top_and_flop
      IMPORTING
        VALUE(iv_number) TYPE i
      EXPORTING
        VALUE(et_top)    TYPE tt_bupa_selection
        VALUE(et_flop)   TYPE tt_bupa_selection.

  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_oia_topandflop IMPLEMENTATION.

  METHOD get_top_and_flop BY DATABASE PROCEDURE
                            FOR HDB
                            LANGUAGE SQLSCRIPT
                            OPTIONS  READ-ONLY
                            USING snwd_so_i snwd_so snwd_bpa.

    --retrieve the best customers
    et_top = select top :iv_number bp.company_name as company_name, sum(soi.gross_amount) as gross_amount
               from snwd_so_i as soi
                inner join snwd_so  as so on so.node_key = soi.parent_key
                inner join snwd_bpa as bp on bp.node_key = so.buyer_guid
                group by company_name
                order by gross_amount desc;

    --retrieve the worst customers
    et_flop = select top :iv_number bp.company_name as company_name, sum(soi.gross_amount) as gross_amount
                from snwd_so_i as soi
                 inner join snwd_so  as so on so.node_key = soi.parent_key
                 inner join snwd_bpa as bp on bp.node_key = so.buyer_guid
                 group by company_name
                 order by gross_amount asc;

  ENDMETHOD.

ENDCLASS.
```

Step 4: Create an ABAP Report consuming the AMDP method

We will now create and implement a simple ABAP report which call the AMDP method and display the result sets on the screen.

Info: An AMDP method is called like any other method in ABAP Objects. This requires, however, that the central database of the current AS ABAP is managed by the database system for which the AMDP method is implemented – meaning SAP HANA in our case. If not, a runtime error is produced. Detailed analysis of such error can be done using the tools of the ABAP Dump Analysis (`ST22`).

1. Create the ABAP program `ZR_OIA_TOPANDFLOP`.
Select the package of your choice, right-click on it and choose context menu entry `New > ABAP Program`.

Enter the required information (name, a description – e.g. “`Retrieve and Display Sales Order with Open Days and BuPa Data`”-) and press Next.

Select a transport request if required and press `Finish` to confirm the creation dialog.

2. Now implement the report.
For this purpose, just copy & paste the source code below into the ABAP editor.

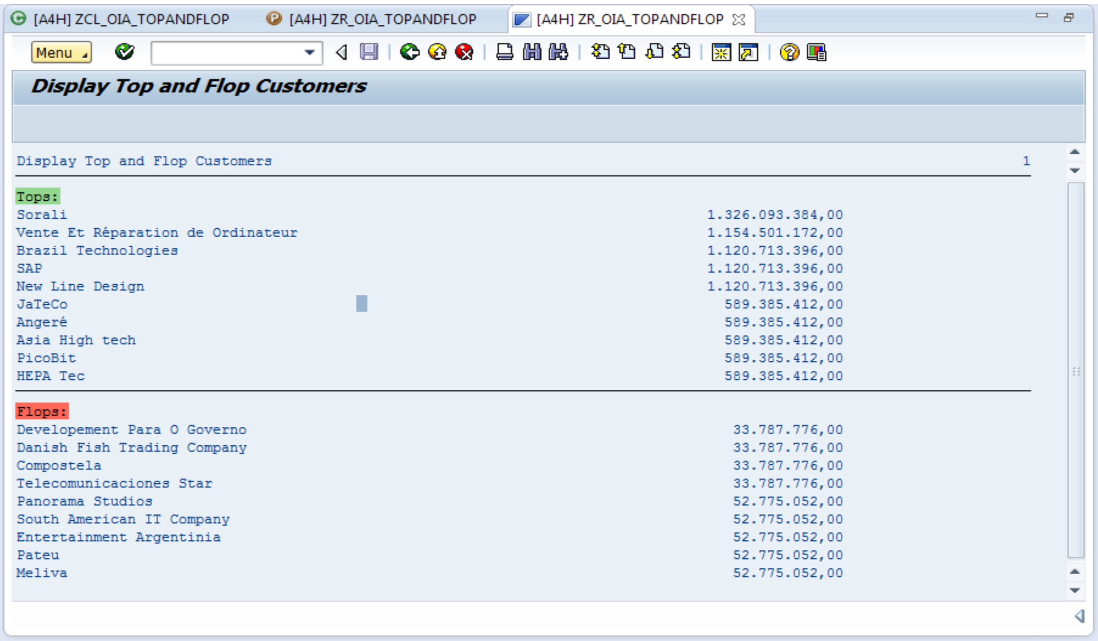
```
PARAMETER pnumber TYPE i DEFAULT 10.

DATA: lv_number TYPE  i.

* set the value of the procedure input parameter
lv_number = pnumber.
* call  AMDP methods
zcl_oia_top_and_flop=>get_top_and_flop(
                EXPORTING iv_number = lv_number
iv_client = sy-mandt
                IMPORTING et_top = data(lt_top)
                          et_flop = data(lt_flop) ).
* display the returned itab with TOP customers
WRITE: / ‘Best customers:’ COLOR COL_POSITIVE.
LOOP AT lt_top ASSIGNING FIELD-SYMBOL(<f>).
  WRITE:/ <f>-company_name , <f>-gross_amount.
ENDLOOP.
* display the returned itab with FLOP customers
ULINE.
WRITE: ‘ Worst customers:’ COLOR COL_NEGATIVE.
LOOP AT lt_flop ASSIGNING FIELD-SYMBOL(<g>).
  WRITE:/ <g>-company_name , <g>-gross_amount .
ENDLOOP.
```

3. Save and activate your test report.

4. You can now run the report (press `F8`) and see the result of your effort.



```
Display Top and Flop Customers                                                      1

Tops:
Sorali                                                          1.326.093.384,00
Vente Et Réparation de Ordinateur                               1.154.501.172,00
Brazil Technologies                                             1.120.713.396,00
SAP                                                             1.120.713.396,00
New Line Design                                                 1.120.713.396,00
JaTeCo                                                            589.385.412,00
Angeré                                                            589.385.412,00
Asia High tech                                                    589.385.412,00
PicoBit                                                           589.385.412,00
HEPA Tec                                                          589.385.412,00

Flops:
Developement Para O Governo                                        33.787.776,00
Danish Fish Trading Company                                        33.787.776,00
Compostela                                                         33.787.776,00
Telecomunicaciones Star                                            33.787.776,00
Panorama Studios                                                   52.775.052,00
South American IT Company                                          52.775.052,00
Entertainment Argentinia                                           52.775.052,00
Pateu                                                              52.775.052,00
Meliva                                                             52.775.052,00
```

Summary

Congratulations! You have just experienced how easy it is to implement an SQLScript-based AMDP and consume it in ABAP.

Regarding the transport aspect, AMDP classes are transported similarly to regular ABAP classes using the standard ABAP transport mechanism. No HANA artifacts have to be transported. This means that the HANA transport container and the HANA delivery units are not involved in this process.

You can have a look at video tutorials demonstrating how to create an AMDP and how to debug an AMDP here.

Tipp: As direct comparison to this “Top-Down”-approach, you can have a look at another step-by-step tutorial showing the “Bottom-Up”-approach for consuming the same HANA database procedures using an ABAP database procedure proxy.

Related Content

- Find more content under `ABAP for SAP HANA`,SCN space
- `ABAP for SAP HANA Reference Scenario`, SCN document
- `SAP HANA SQLScript Reference`, SAP Help Guide

Like Alert Moderator

Assigned tags

ABAP Development | abap for hana | former jive document | hana | sap |

View more...

Related Blogs

An example of AMDP(ABAP Managed Database Procedure) in 740

By **Jerry Wang** , Dec 10, 2013

Consuming SAP HANA Procedures from ABAP [In Side Car Approach]

By **Former Member** , Sep 25, 2014

ABAP for SAP HANA Reference Scenario – Tutorials

By **Carine Tchoutouo Djomo** , Oct 23, 2014

Related Questions

ABAP Object services and Hana

By **Daniel Mueller** , Dec 21, 2016

Issue when processing the AMDP - Dump -- AMDP_UNCOMMITTED_DBPROC_CALLED

By **Former Member** , Mar 30, 2017

HANA Studio AIE ABAP Development Plugin - Can't create a New Project

By **Former Member** , Aug 23, 2017

25 Comments

You must be Logged on to comment or reply to a post.

Vivek RR

March 18, 2014 at 2:02 pm

Hi Carine

Nice blog..keep up the good work. have a question though.. Assuming we are working on code pushdown for an existing code, and my logic involves having the input for my pushdown logic as an internal table ..it should be possible in providing it as an input parameter, right ?

Like (0)

Carine Tchoutouo Djomo | Post author

March 19, 2014 at 11:49 am

Hi Vivek,

Thanks for your feedback!

Yes, you can provide input parameters of type table.

Procedures can have multiple input parameters and output parameters (which can be of scalar types or table types). My example above defines a read-only procedure which has 1 scalar input parameter and 2 output parameters of type table.

You can find more information in the SAP HANA SQLScript Reference Guide (e.g. page 16)

Kind regards,

Carine

Like (0)

Vivek RR

March 19, 2014 at 2:17 pm

hi carine

Thanks again..

I know we can provide input as table but not sure if we map directly the structure of of internal table from ABAP later. Consider this ..my abap logic has an internal table which has 30 columns and I want to use that as input to my procedure. So should I should create table type in HANA manually (create type as table t1 (col1 type , col2….col30)) OR I can point directly to a table in ABAP as we declare internal table?

Like (0)

Jasmin Gruschke

March 19, 2014 at 2:53 pm

Hi Vivek,
that depends on what you would like to do with the input table later on in your SQLScript procedure. You could e.g. think of just selecting from the input table via "SELECT <field> from :<input_table>", then you don't need the type. Furthermore, you already get transfer tables defined, just execute the AMDP once and search in the HANA Catalog (schema SAP<SID>) in "Tables" and search for "CLASSNAME"=>"AMDP_METHOD_NAME". You will find xxx#tft tables there, i.e. transfer tables.

Cheers,

Jasmin

Like (0)

Vivek RR

March 19, 2014 at 3:11 pm

Oh..great ..let me check that then. thanks a lot Jasmin

Like (0)

Former Member

April 9, 2014 at 5:34 pm

Hi,

I’m trying to implement the _DBSYS_OPT model. I have defined an interface with my AMDP methods and I’m trying to call one of these methods inside another AMDP method. The problem is I can’t figure out the proper syntax for the USING clause to specify a method defined in an interface. To be clear, both methods are defined in the same interface, implemented in same class, and both methods are AMDP methods.

When I wasn’t using an interface method syntax like this works fine:

```
METHOD my_method_name BY DATABASE PROCEDURE
 FOR HDB LANGUAGE SQLSCRIPT
 OPTIONS READ-ONLY
 USING table1
     table2
     table3
     my_class=>my_other_method .

<implementation details>

ENDMETHOD.
```

This also works fine when my_method is defined in the interface but only tables are in the USING clause. Like:

```
METHOD my_interface~my_method_name BY DATABASE PROCEDURE
…
```

After defining the interface the syntax I’ve tried is;

```
METHOD my_interface~my_method_name BY DATABASE PROCEDURE
  FOR HDB LANGUAGE SQLSCRIPT
  OPTIONS READ-ONLY
  USING table1
        table2
        table3
        my_class->my_interface~my_other_method .

<implementation details>

ENDMETHOD.
```

This results in an error:

Invalid character string “->” in “MY_CLASS->MY_INTERFACE~MY_OTHER_METHOD”. Only transparent database tables, views, or database procedures can be specified after USING.

I’ve also tried a static reference like:

```
my_class=>my_interface~my_other_method .
```

The resulting error is:

The user declaration for the object “MY_CLASS=>MY_INTERFACE~MY_OTHER_METHOD” is not needed.

So I try removing the method from the USING clause and where the method is called I get an MY_CLASS=>MY_INTERFACE~MY_OTHER_METHOD is unknown, with or without the inteface.

I’ve tried defining an alias like:

```
INTERFACES my_interface.

ALIAS my_other_method FOR my_interface~my_other_method .
```

and then using

```
my_class=>my_other_method .
```

Then I get a message like:

```
“MY_CLASS=>MY_INTERFACE~MY_OTHER_METHO” expected, not
“MY_CLASS=>MY_INTERFACE~MY_OTHER_METHOD” (see long text).
```

I’m only having an issue with methods defined in the interface when I try to use another method, defined in the same interface/class, in the USING clause.

Does anyone know the proper syntax when using an interface defined method in the USING clause of an AMDP procedure?

Thanks, George

Like (0)

Former Member

April 9, 2014 at 6:01 pm

Hi,

I must have had a misspelling in either the USING clause or the method call. I asked my coworker to take a look and he copied the interface/method name from the definition, prepended the class name using the static reference operator, pasted in both the USING clause and method call. Like:

```
MY_CLASS=>MY_INTERFACE~MY_OTHER_METHOD
```

No errors.

Lesson learned: try the simple things first.

Cheers, George

Like (0)

Former Member

October 29, 2014 at 2:08 am

thanks

Like (0)

Former Member

November 19, 2014 at 5:37 am

Thanks. Good info.

Like (0)

Sreehari V Pillai

January 5, 2015 at 9:18 am

Nice document Carine Tchoutouo Djomo .

Sree

Like (0)

Vipin Nagpal

April 27, 2015 at 10:30 am

Hi Carine,

I want to share information regarding SQL script. I am new to SQL script and i am having a beginner level.

Just notice one aspect of SQL script. If i do not pass CLIENT in where condition of SQL script, then i am getting sum of gross amount of all clients.

Thanks

Like (0)

Sreehari V Pillai

April 27, 2015 at 10:37 am

yes Only graphical calculation views default the client from session client .

Like (0)

Carine Tchoutouo Djomo | Post author

April 29, 2015 at 8:50 am

Hi Vipin,

That's correct. Analog to native SQL with ADBC and EXEC, the AMDP framework does not support automatic client handling. I'll add this info to the text above.

If the computation is client-specific, then the client context has to be passed to the AMDP method and used appropriately in the SQLScript coding.

Example:

```
— AMDP method definition

CLASS-METHODS get_top_and_flop

  IMPORTING

    VALUE(iv_client) TYPE mandt

    VALUE(iv_number) TYPE i



  EXPORTING

    VALUE(et_top)     TYPE tt_bupa_selection

    VALUE(et_flop)   TYPE tt_bupa_selection.



— SQLScript: retrieve the best customers

et_top = select top :iv_number bp.company_name as company_name, sum(soi.gross_amount)
as gross_amount

          from snwd_so_i as soi

          inner join snwd_so  as so
            on  so.node_key = soi.parent_key
            and so.client  = soi.client

          inner join snwd_bpa as bp
            on  bp.node_key = so.buyer_guid
            and bp.client  = so.client

          where soi.client = :iv_client

          group by company_name

          order by gross_amount desc;
```

Kind regards,

Carine

Like (0)

Vipin Nagpal

April 29, 2015 at 10:15 am

Thanks a lot for your reply. You gave correct sample code, which i was looking for .

Like (0)

Vipin Nagpal

May 22, 2015 at 10:30 am

Hi Carine,

What will happen if i want to add currency conversion on above demo example.

How the code will look like? We should not ignore currency while adding gross amount.

Thanks

Like (0)

Jasmin Gruschke

May 27, 2015 at 12:02 pm

Hi Vipin,
you're right, we should not leave out currency conversion here.
What we would like to emphasise is the understanding of the concepts of AMDPs, rather than the implementation details for SQLScript procedures. Of course this is very important for thos implementing the AMDPs in a real-live application, but therefore, you have to get familiar with – or even better you should become an expert for – SQLScript.

You can find an example for currency conversion (via CE functions) on page 25 of http://scn.sap.com/docs/DOC-54541 of in the ABAP language documentation ABAP Keyword Documentation.
If you prefer SQL functions see CONVERT_CURRENCY – SAP HANA SQL and System Views Reference – SAP Library (an example is given at the very end of the page).

Cheers,
  Jasmin

Like (0)

Vipin Nagpal

May 28, 2015 at 5:20 am

Thanks a lot Jasmin for your guidance.

I think if we talk about aggregate function of data, then conversion is equally important before applying aggregate function on data.

Like (0)

Mainak Aich

September 14, 2017 at 11:01 pm

Hello Jasmin,

The document you referred at http://scn.sap.com/docs/DOC-54541 doesn't exists anymore. Could you please check and provide this again(maybe a updated version of the document with new features of ABAP 7.51 😉).

Thanks,

Mainak

Like (0)

Former Member

January 27, 2016 at 6:14 am

Very good work...Thanks.

Like (0)

Former Member

May 19, 2016 at 11:35 am

Hi Carine

nice blog !

I have a question

can we call simple method from amdp method .

if yes then how to call it .

Like (0)

Former Member

June 7, 2016 at 8:59 am

Hi Carine and other experts,

I have one query, I checked one of the AMDP class and method, and executed the same from ABAP through the normal SAP GUI. Can you pls let me know once we have executed the method, the DB procedure gets created and gets called right ?

No I wanted to know how do I view this DB procedure in HANA through the studio? In which schema of 'Catalog' does it get into?

Thanks in advance.

Like (0)

Amol Samte

June 7, 2016 at 10:43 am

Hi,

Once you run AMDP first time it will get created on HDB or will get overwrites if already available...

Now where to see the procedure in HDB —

1. HANA Modeler(Log in HDB) – Catlog – SAP Schema(Name of your SAP schema) – Procedures..

-Amol S

Like (0)

Rajarshi Muhuri

December 21, 2016 at 11:08 pm

Hi Carine

I was trying to write a AMDP where the SQL method is reading from a ABAP schema.

```
METHOD MY_HANA_PROCEDURE BY DATABASE PROCEDURE
FOR HDB
LANGUAGE SQLSCRIPT
OPTIONS READ-ONLY
USING SAPABAP1.ZTSQL_DATA20.

E_VIEW = SELECT CALMONTH2, CALYEAR, Z_PROD, Z_AMOUNT, Z_QUAN   FROM SAPABAP1.ZTSQL_DATA20;

ENDMETHOD.
```

even though the column table ZTSQL_DATA20 exists, and can be directly queried by SQL – I am getting an error that says

“Only Transparent tables , database views, and procedures can be used”

Not sure why I am getting the error.

also is there a way I can read calculation views of HANA within a AMDP (top down approach) . The hana views are stored in _SYS_BIC schema .

finally how do i read tables that are in form /BIC/CALMONTH2 .

In HANA SQL , I wrap them in “/BIC/CALMONTH2” but I guess the systax is different here

Like (0)

Rajarshi Muhuri

December 21, 2016 at 11:36 pm

I realized my mistake .. I was probably getting an error as that table was created thru the HANA side , so even thought the table exist in the abap schema , its not abap managed as I could not see the table via se16 .

So that answers my second question that I wont be able to use HANA calculation views , but would have to re write them as CDS views and then be able to use them .

Like (0)

Former Member

March 10, 2017 at 2:10 pm

Hi,

I think you may be able to help me. In my case I have a table in BW. I want to access to this table, pre-process it and apply a AFL predictive function (*Single Exponential Smoothing*). Then, I want to store the results and access them from BW. After some research, I managed to find 5 different ways to approach this problem:

1. Using SQLScript from SAP HANA Studio
2. Using Application Function Modeller (AFM) from SAP HANA Studio
3. Using SAP Predictive Analytics (SAP PAA)
4. Use BW report (ABAP) to trigger a SQL PROCEDURE
5. Using Predictive Algorithms native in BW

With a college help, we created an ABAP code that calls my SQLScripts. With the ABAP we can also pre-process the data and create all necessary tables required to run the SQLScripts. On the one hand we failed to run the SQLScript approach #1. We tried different variations in the CALL sentence with any success. On the other hand, we succeed when using the same ABAP call but using the CALL of the AFM approach #2. The problem might be related to the CALL parameters or to the use of “*With Overview / In Debug Mode*“.

Please, have a look to https://answers.sap.com/questions/148420/five-approaches-to-execute-a-predictive-afl-functi.html.

Thanks and regards!

Like (0)

Share & Follow

Privacy

Legal Disclosure

Trademark

Sitemap

Terms of Use

Copyright

Cookie Preferences

Newsletter