Technical Articles

Santhosini K

April 14, 2021 | 5 minute read

SQL Script for ABAP Managed Database Procedures(AMDP)-Code pushdown for a better performance!

Follow

10 44 6,366

Like

RSS Feed

We all are aware of the term “**code pushdown**” in the SAP HANA database and how it helps us in improving the performance of the application.

When it comes to performance intensive applications say an analytical report, the bottleneck lies in moving the records between the database server and the application server. The time taken is directly proportional to the number of records moved between the database server and the application server.

We all are used to the technique of fetching the records using CDS views and make other calculations/processing/filtration in the ABAP layer.

Here I am referring to both the flavors of CDS views – SAP ABAP CDS views and the external views generated from the SAP HANA CDS views.

The idea here is to perform all the processing of records in the database layer, rather than moving the large amount of unprocessed records to the ABAP layer.

When it comes to CDS views , we face certain limitations in terms of processing the data the way we want . Examples are delete the adjacent duplicates or use of Order by clause. That’s when we think of Table functions in SAP HANA using ABAP Managed Database Procedures(AMDP) as a savior.

Since the Table functions are built using SQL Script they offer a lot of flexibility to code simple to complex logic

Here is a handy SQL Script guide for the basic operations those we perform in the ABAP layer in order to process the data the way we want

Please Note: Use the AMDP table functions only in places where you cannot use the CDS views. CDS views are preferred over AMDP table functions for the optimization and parallelization they offer.

Simple operations.

1. Declare internal table inside AMDP class
2. Declare an ABAP datatype in SQL script
3. Delete adjacent duplicates
4. Sort by column and pick the latest value
5. Convert a delimited string to an internal table
6. Apply filter to local table
7. Calling AMDP methods with parameters
8. Check if the Internal table is not initial
9. Select client specific data inside the AMDP method
10. Convert the rows to columns using "Case statement" (Transposition)

Declare internal table inside AMDP class

Go to the AMDP class and declare the internal table in the public section. Here we can make use of the ABAP syntax and the ABAP datatypes. Declaring the global table types are helpful in calling the AMDP methods with return parameters.

```
class zcl_com_final definition
 public
 final
 create public.
  public section.
    interfaces if_amdp_marker_hdb.
    types: begin of ty_itab,
         rownum        type int2,
             db_key        type /bobf/conf_key,
             prod_hr_id    type /dmf/hierarchy_id,
            creation_date  type dats,
end of ty_itab,
gt_itab type standard table of ty_itab with unique key primary_k
```

Declare ABAP data type inside the SQL script

Below is an example of how we can declare an ABAP specific data type inside the AMDP method using the SQP script

```
declare lv_timestamp   "$ABAP.type( TZNTSTMPS )";
```

Delete adjacent duplicates

“Delete adjacent duplicates” is a very common statement in ABAP. Below is the syntax for the same in SQL script. This statement deletes the adjacent duplicate records based on the field “db_key” from table lt_itab

```
Lt_itab_noduplicates =  SELECT * FROM (  select
                        row_number() over ( partition by db_key
                        :lt_itab) where rownum = 1 ;
```

Sort by column and pick the latest value

This is one stellar operation that we cannot achieve with our traditional CDS views. This is one of the most useful statements when it comes to filtering of the unwanted records

The below statement picks the latest offer number for the given product group id.

```
lt_latestoffer = select * from ( select row_number() over ( par
                 order by creation_date desc ) as  rownum , * fr
```

Here is the sample data

EY	Prod_hr_id	Date
	123	4/4/2021
	123	4/5/2021
	123	4/6/2021

	456	4/7/2021
	456	4/8/2021
	456	4/9/2021

Output:

DB_KEY	Prod_hr_id	Dat
3	123	4/6
6	456	4/9

Convert a delimited string to an internal table

CDS views do not support a larger string operation. The string functions are not supported for the datatype “STRING” . The below chunk of code comes handy when we have to pass multiple values as a parameter to the table function and later split them and use them inside the AMDP method.

Assume the value in lv_string = ABC|DEF|GHI|JKL

```
split_values = SELECT substr_before(:lv_string,'|') single_val 
      SELECT substr_after(:lv_string,'|')  INTO lv_string FROM 
         while( length(:lv_string) > 0 )
            DO
                 split_values = SELECT substr_before(:lv_string
                     UNION
                  SELECT single_val FROM :split_values;
                  SELECT substr_after(:lv_string,'|') INTO lv_s
            END while;
   itab = SELECT single_val AS "OUTPUT_SPLIT" FROM :SPLIT_VALUE
```

Itab :

OUTPUT_SP
ABC
DEF

GHI
JKL

Apply filter to a local table

I have made this example with product group number but in real time this can be used to separate the process types or any particular group of data from the other

ITAB

DB_KEY	Prod_hr_id	Date
3	123	4/6/202
6	456	4/9/202

```
declare lc_filter  string  := '( PROD_HR_ID = ' || '''123''' ||
itab_result = apply_filter ( :itab , :lc_filter );
```

ITAB_RESULT

DB_KEY	Prod_hr_id	Date
3	123	4/6/202

Calling AMDP methods with parameters

We can have an AMDP method with import and export parameter. This helps in modularizing and reusing the code.

Declare the class method like this

I have declared it with one importing parameter and one exporting parameter. You can have multiple import and export parameters to support your programming logic.

```
public section.
class-methods:
      get_ofrmain
```

```
        importing
          value(p_adzone)  type  char255
        exporting
          value(et_ofrmain)   type gt_itab.
```

Calling *get_ofrmain* method inside another method *ofr_adzone*.

```
method ofr_adzone
      by database function
      for hdb
      language sqlscript
      options read-only
      using  zcl_com_final=>get_ofrmain.

call "ZCL_COM_FINAL=>GET_OFRMAIN"  ( P_ADZONE => :P_ADZONE ET_OI
LT_OFRMAIN = SELECT * FROM :ET_OFRMAIN;
```

Check if Internal table is not initial

This is one important statement in our ABAP programing model and the most frequently used statement

```
SELECT COUNT(*) INTO numrows FROM :LT_OFRMAIN;
   IF numrows > 0 then
// program logic
   END IF;
```

Select the client specific data

Its very important to select client specific data while working with database schemas. The below method selects client specific data from a Z table ZPRD_DEPT which is part of the schema SAP_S4HANA

```
method Prd_dept
      by database function
      for hdb
      language sqlscript
      options read-only.
    RETURN select _Prd.mandt  as clnt, _Prd.sfs_dept_num,_Prd.s
```

```
              from "SAP_S4HANA"."ZPRD_DEPT" as _Prd where _Prd.ma
  endmethod.
```

Convert the Rows to Columns using "Case Statement" (Transposition)

This operation is not supported in the CDS when the given datatypes are of "STRING". During such instances , instead of jumping into the ABAP layer , we can efficiently perform such operations using SQL Script in AMDP table functions.

ZPRD_ATTR

PRODHRID	ATTRIBUTE	ATTRIBUTEVA
123	0001	COLOR : YELL(
123	0002	SIZE : 10 GRAI
123	0003	TYPE : JELLY
456	0001	COLOR : BLUE
456	0002	SIZE : 500 GRAMS
456	0003	TYPE : CREAM

```
method get_prodatt
        by database function
        for hdb
        language sqlscript
        options read-only
        using zprd_attr.

    lt_att = select prodhrid,

             max (case
             when attribute = '0001' then
             cast(attributevalue as char( 255 ))
             end ) as ATTRIBUTEVALUE1,

            max (case
              when attribute = '0002' then
              cast(attributevalue as char( 255 ))
              end ) as ATTRIBUTEVALUE2,

              max (case
              when attribute = '0003' then
              cast(attributevalue as char( 255 ))
              end ) as ATTRIBUTEVALUE3

              from zprd _attr
              group by prodhrid;

 return select  prodhrid , concat(  ATTRIBUTEVALUE1, concat(ATTRI
        as Ovrline  from :lt_att;

   endmethod.
```

OUTPUT

PRODHRID	OVERLINE
123	COLOR : YELLOW SIZE : 10 GRAMS TYPE: JELLY
456	COLOR : BLUE SIZE : 500 GRAMS TYPE : CREAM

I have extensively worked on the performance optimization of fiori applications using code push down. I shall talk about the performance optimization techniques for CDS views and Table functions in my next blog post.

Try using these SQL scripts in the AMDP classes instead of using the ABAP layer and do let me know if this made your application run faster.

Incase you have a better way of doing this , I am all ears.

Cheers,

Santhosini K

Alert Moderator

Assigned tags

SAP HANA

ABAP Development

SQL

ABAP AMDP

codepushdown

HANA SQLScript

Similar Blog Posts

Understanding evolution of CDS and AMDP in most simple way

By Ujjwal Singh May 19, 2019

Code-to-Data Paradigm in ABAP with SAP HANA

By Srinivas Gadilli Mar 14, 2020

AMDP - ABAP consume Hana SQL / View Directly - Part I

By Venkateswaran (Venkat) Krishnamurthy Mar 09, 2020

Related Questions

HANA AMDP vs Regular ABAP runtime?

By Former Member Feb 04, 2015

Is it good to learn PL/SQL before starting with SAP HANA SQL script?

By Prakash K Nov 22, 2018

AMDP Select Statement

By Former Member Oct 23, 2015

10 Comments

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

Vigneswaran Mathivanan
April 14, 2021 at 10:49 am

Excellent Santhosini...Looking forward to few more blogs related to this topic..

Like 0 | Share

Rajiv Kanoria
April 14, 2021 at 3:01 pm

Nice Blog Sathosini

Like 0 | Share

AjeethKumar R
April 14, 2021 at 7:18 pm

Excellent blog!!..

Like 0 | Share

Harry Jing
April 15, 2021 at 12:53 am

The blog is only used for HANA?   R/3 is not used now ?

Like 0 | Share

Aman Garg

April 15, 2021 at 2:20 am

This is very helpful. Thanks for sharing!

Like 0 | Share

Venkat dattatreya

April 15, 2021 at 5:49 am

Thanks for the blog.

For last use case I think string_agg( ) will also work.

https://help.sap.com/viewer/4fe29514fd584807ac9f2a04f6754767/2.0.04/en-US/a924ee1e98ab435a874efa32e6f0ae14.html

Like 0 | Share

Santhanalakshmi Sankarakrishnan

April 16, 2021 at 7:39 am

Very informative blog. Am sure people can look up to the details whenever they have to implement similar ones in their projects. Keep it up and share more such blogs!

Like 0 | Share

Amelia Scott

April 16, 2021 at 8:24 am

Useful Information, your blog is sharing unique information…
Thanks for sharing!!!

Like 1 | Share

Rahul Abrol

April 21, 2021 at 11:07 am

Thanks for sharing

Like 0 | Share

Rakesh Chandra Joshi

September 6, 2021 at 4:46 pm

Excellent ! Very informative

Like 0 | Share