Sreehari V Pillai
March 18, 2018 6 minute read

# Horrible Performance, HANA push down to the rescue

## Introduction

ABAP on HANA is a hot topic in forums and training sessions these days. We have been seeing lot of general queries raised by (conventional) ABAPers about converting ABAP based business logics to ABAP on HANA using CDS and AMDP. ABAP , being a simple programming language , developers are not familiarize with core SQL functionalities of a database. Well, then its the time to learn them and design perfect programs for your customer.

We all already know , what is CDS and AMDP and why them. Our topic of interests is , how to map the existing abap logic to SQL blocks / CDS statements effectively. Officially , this is a "**Tips and Tricks for beginners**"

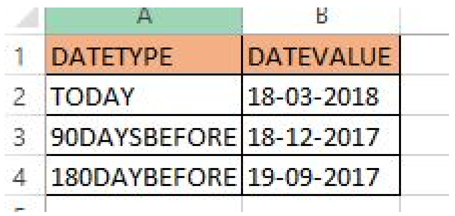
## Motive

Recently we had to touch up an existing ABAP report ( for a multi million dollar business ) which was taking nearly 1 hour to execute ( in background ) . We first , applied basic performance improvements to it from ABAP level and brought down the execution time to 20-25 Minutes . Then , pushing down core business logics to database ( using CDS & Table functions ) the execution time came down to less than a second.

**Disclaimer** : Logics explained below may not be the best performing solution. But , some idea on how all we can look into a case and implement them in ABAP on HANA.

## 1. View with list of hard coded values

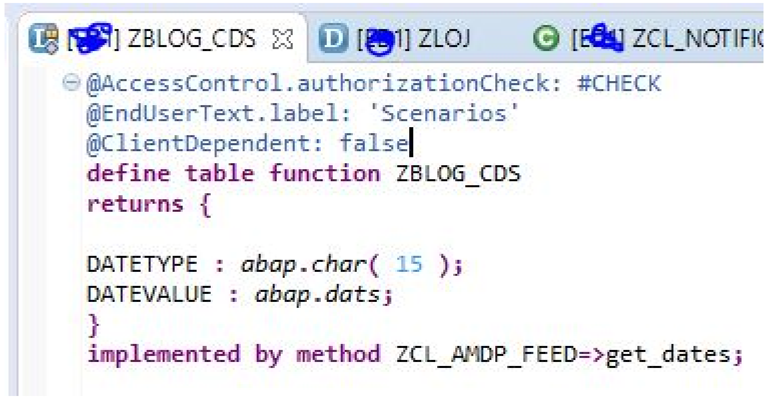
Requirement : Generate a view which returns fixed values based on a condition. For example, I need a CDS view which returns data as below.



| | A | B |
|---|---|---|
| 1 | DATETYPE | DATEVALUE |
| 2 | TODAY | 18-03-2018 |
| 3 | 90DAYSBEFORE | 18-12-2017 |
| 4 | 180DAYBEFORE | 19-09-2017 |

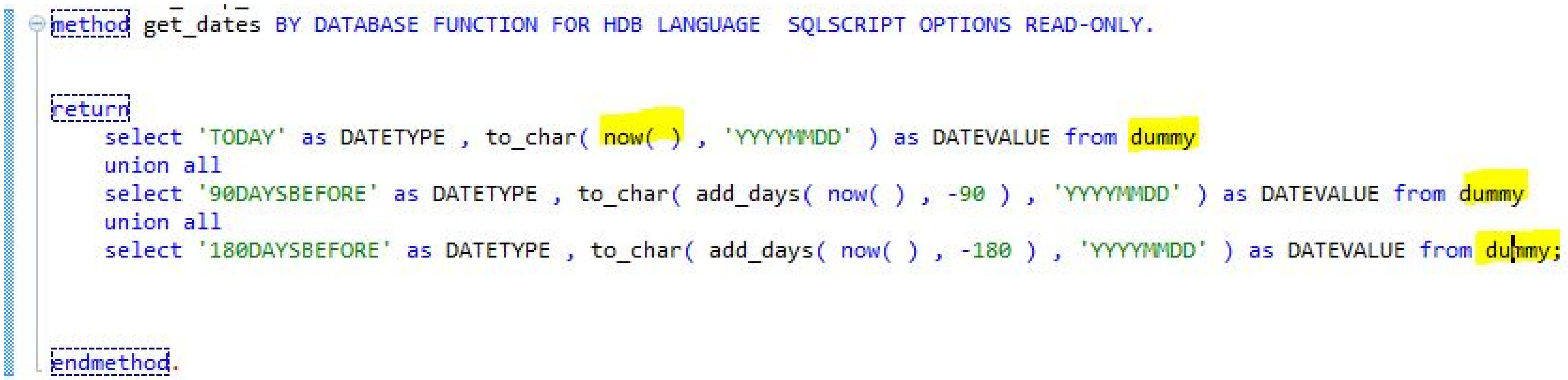
There is no DDIC tables involved in this scenario to query and get the result.

Lets create a table function and associate it with an AMDP method.  refer this link



```
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Scenarios'
@ClientDependent: false
define table function ZBLOG_CDS
returns {

DATETYPE : abap.char( 15 );
DATEVALUE : abap.dats;
}
implemented by method ZCL_AMDP_FEED=>get_dates;
```

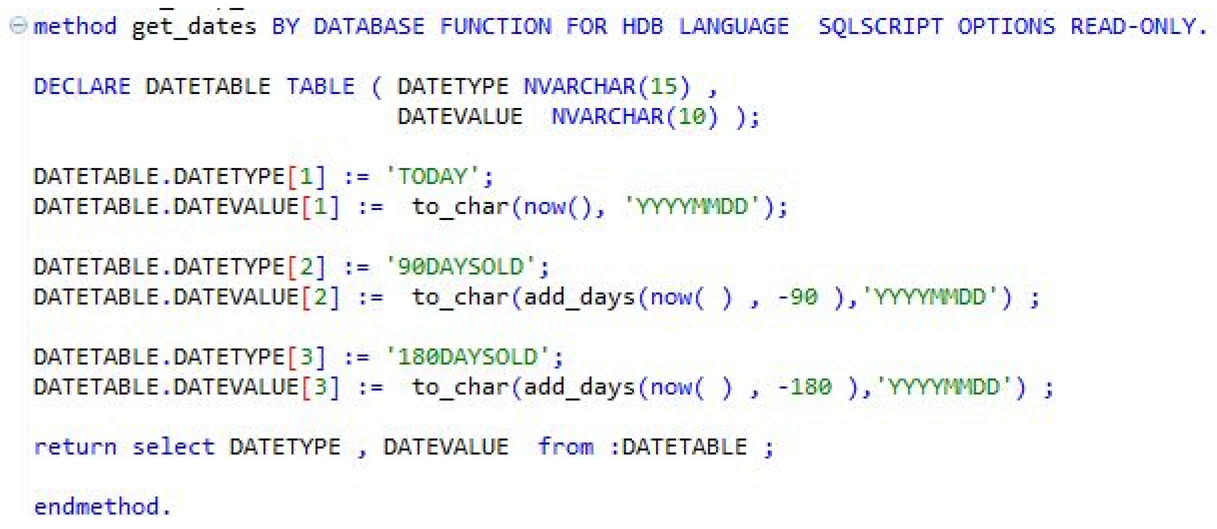
and Table function implemented as



```
method get_dates BY DATABASE FUNCTION FOR HDB LANGUAGE  SQLSCRIPT OPTIONS READ-ONLY.


return
    select 'TODAY' as DATETYPE , to_char( now( ) , 'YYYYMMDD' ) as DATEVALUE from dummy
    union all
    select '90DAYSBEFORE' as DATETYPE , to_char( add_days( now( ) , -90 ) , 'YYYYMMDD' ) as DATEVALUE from dummy
    union all
    select '180DAYSBEFORE' as DATETYPE , to_char( add_days( now( ) , -180 ) , 'YYYYMMDD' ) as DATEVALUE from dummy;



endmethod.
```

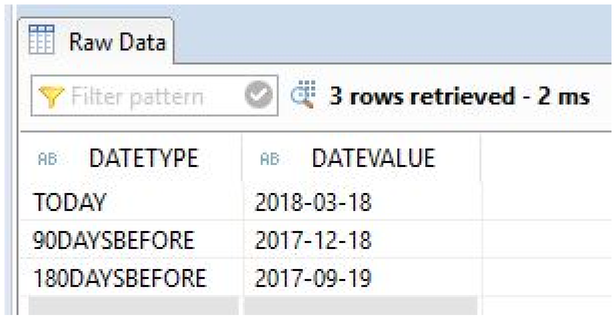
Have you noticed the usage of table dummy in the where clause and not declared using the "**using**" clause in method signature ? dummy is not a DDIC object and not a normal  database table. DUMMY is a system table , part of the language construct and *has always exactly one row*. We can use dummy to perform this type of functionalities. The usage of built-in functions above is self explanatory .

Alternately using  local table variable:



```
method get_dates BY DATABASE FUNCTION FOR HDB LANGUAGE  SQLSCRIPT OPTIONS READ-ONLY.

DECLARE DATETABLE TABLE ( DATETYPE NVARCHAR(15) ,
                          DATEVALUE  NVARCHAR(10) );

DATETABLE.DATETYPE[1] := 'TODAY';
DATETABLE.DATEVALUE[1] :=  to_char(now(), 'YYYYMMDD');

DATETABLE.DATETYPE[2] := '90DAYSOLD';
DATETABLE.DATEVALUE[2] :=  to_char(add_days(now( ) , -90 ),'YYYYMMDD') ;

DATETABLE.DATETYPE[3] := '180DAYSOLD';
DATETABLE.DATEVALUE[3] :=  to_char(add_days(now( ) , -180 ),'YYYYMMDD') ;

return select DATETYPE , DATEVALUE  from :DATETABLE ;

endmethod.
```

**Output :**

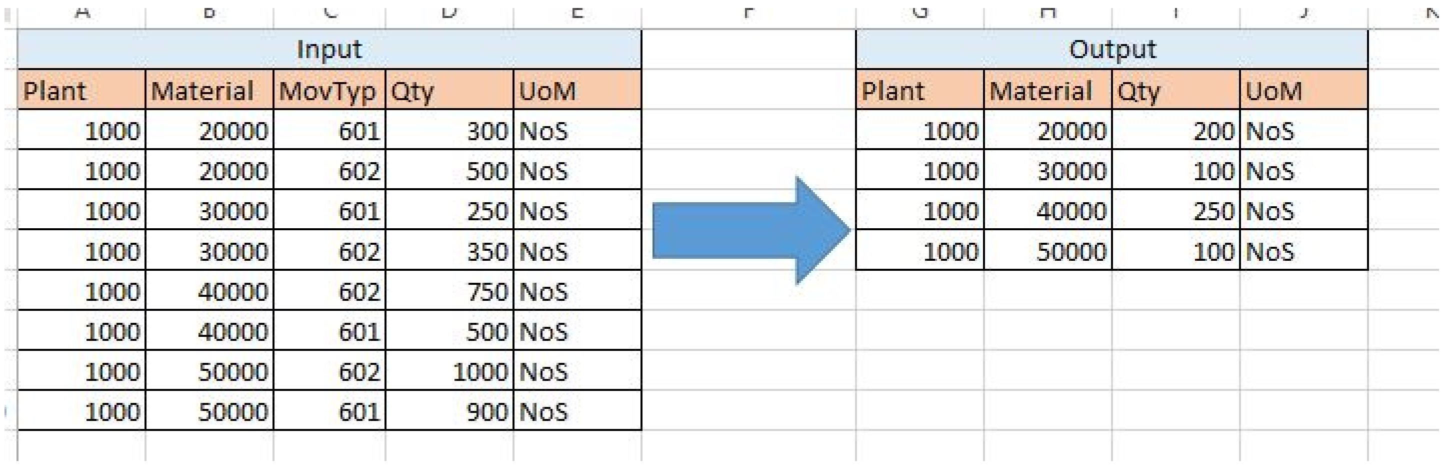


3 rows retrieved - 2 ms

| DATETYPE | DATEVALUE |
|---|---|
| TODAY | 2018-03-18 |
| 90DAYSBEFORE | 2017-12-18 |
| 180DAYSBEFORE | 2017-09-19 |

## 2 . Aggregating columns conditionally

In the below example,  the requirement is to Plant wise , Material wise stock data for movement types 601 and 602. Condition is , 601 movement type quantity should be subtracted from 602 quantity while aggregating.



| Input | | | | |
|---|---|---|---|---|
| Plant | Material | MovTyp | Qty | UoM |
| 1000 | 20000 | 601 | 300 | NoS |
| 1000 | 20000 | 602 | 500 | NoS |
| 1000 | 30000 | 601 | 250 | NoS |
| 1000 | 30000 | 602 | 350 | NoS |
| 1000 | 40000 | 602 | 750 | NoS |
| 1000 | 40000 | 601 | 500 | NoS |
| 1000 | 50000 | 602 | 1000 | NoS |
| 1000 | 50000 | 601 | 900 | NoS |

| Output | | | |
|---|---|---|---|
| Plant | Material | Qty | UoM |
| 1000 | 20000 | 200 | NoS |
| 1000 | 30000 | 100 | NoS |
| 1000 | 40000 | 250 | NoS |
| 1000 | 50000 | 100 | NoS |

Corresponding ABAP logic could be, to send 2 separate SELECT requests for 601 movement type and 602 movement type aggregated based on the Plant , Material and Quantity and then loop over first result set and read second . Then perform the arithmetic operation to arrive at the expected output. Or may be, using new OPEN SQL syntax, add a calculated column based on the movement type and later collect the records in application layer.

Look at the below statement.

```
return select WERKS as PLANT,
       MATNR as MATNR,
       sum(
         case
                         when BWART = '601' then -1 * MENGE
                         when BWART = '602' then  MENGE
                end
          ) as STOCK ,
        MEINS as UOM
       from MSEG where BWART in ( '601' , '602' )
       GROUP BY WERKS, MATNR , MEINS
       ;
```

Based on the Movement type ( BWART ) , the Menge ( quantity ) field is negated and summated group by plant , material and Unit of measurement .

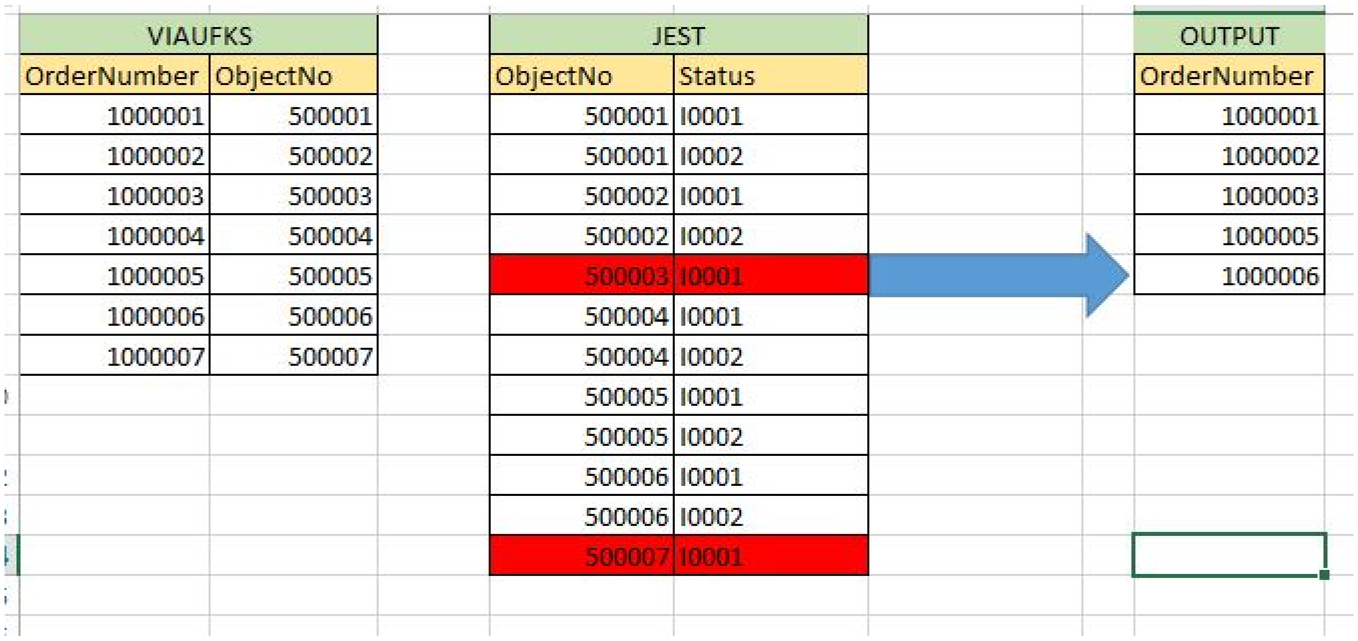
If you really love ABAP, don't let it suffer. Let the DB suffer !

## 3. Logic Block – Fetch entries having 2 statuses

In Plant maintenance , the PM Order view VIAUFKS  and Object status table JEST have huge number of records. In the below example, requirement is to fetch orders which has 2 statuses in the order table.

Say, wee need to fetch all orders created this month having 2 status entries in JEST table ‘I0001’ and ‘I0002’ ;

Sample Data and expected output as below.



| VIAUFKS | |
|---|---|
| OrderNumber | ObjectNo |
| 1000001 | 500001 |
| 1000002 | 500002 |
| 1000003 | 500003 |
| 1000004 | 500004 |
| 1000005 | 500005 |
| 1000006 | 500006 |
| 1000007 | 500007 |

| JEST | |
|---|---|
| ObjectNo | Status |
| 500001 | I0001 |
| 500001 | I0002 |
| 500002 | I0001 |
| 500002 | I0002 |
| 500003 | I0001 |
| 500004 | I0001 |
| 500004 | I0002 |
| 500005 | I0001 |
| 500005 | I0002 |
| 500006 | I0001 |
| 500006 | I0002 |
| 500007 | I0001 |

| OUTPUT |
|---|
| OrderNumber |
| 1000001 |
| 1000002 |
| 1000003 |
| 1000005 |
| 1000006 |

Here, except order numbers 1000003 & 1000007 , all orders has 2 statuses correspondingly in JEST.

How would you do it in conventional ABAP ?

```
 "Select all the order numbers and Object numbers
SELECT ORDERNUMBER OBJECTNO from VIAUFKAS INTO TABLE T_VIA where CRDATE IN IR_IN_DATE.

SELECT OBJECTNO , STATUS from JEST
INTO_TABLE T_JEST_01
FOR ALL ENTRIES IN T_VIA
where OBJECTNO = T_VIA-OBJECTNO and
STATUS EQ 'I0001'.

SELECT OBJECTNO , STATUS from JEST
INTO_TABLE T_JEST_02
FOR ALL ENTRIES IN T_VIA
where OBJECTNO = T_VIA-OBJECTNO and
STATUS EQ 'I0002'.

"Now loop T_VIA , READ I_JECT_01 , If success, READ I_JEST_02, and If success - Note the Object Number.
```

Both VIAUFKAS and JEST have millions of records and the above code block is costly in terms of performance.

Note : *Above code is the worst way to implement the logic. in normal ABAP itself we can optimize the process. For the sake of explaining, I wrote it like this.*

Lets see , how can we push it down to a better code.

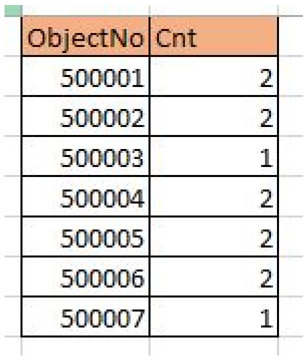
```
 lt_via =        SELECT aufnr ,
                        gewrk ,
                        auart ,
                        objnr ,
                        msgrp ,
                        beber ,
                        gltrs ,
                        qmnum
   FROM viaufks
  WHERE ( gltrs >= :i_date_from and gltrs <= :i_date_to )
   and mandt = :i_mandt;

    lt_via_out = SELECT * from  (  select   v3.objnr as objnr ,
                                   count(*) as cnt
                                  FROM :lt_via v3 inner join jest j
                                 on v3.objnr = j.objnr
                                  AND J.stat IN ('I0001' , 'I0002' )
                                   group by v3.objnr )
  where cnt = 2;
```

**Query 1 :**

it the variable **lt_via**, we are selecting all the order numbers , object numbers and other required fields based on the input dates.

**Query 2 :**

in the subquery ( inside the braces in second sql ) , selecting all the object numbers corresponding to the entries in **lt_via** and status I0001 and I0002 and count of entries .



| ObjectNo | Cnt |
|---|---|
| 500001 | 2 |
| 500002 | 2 |
| 500003 | 1 |
| 500004 | 2 |
| 500005 | 2 |
| 500006 | 2 |
| 500007 | 1 |

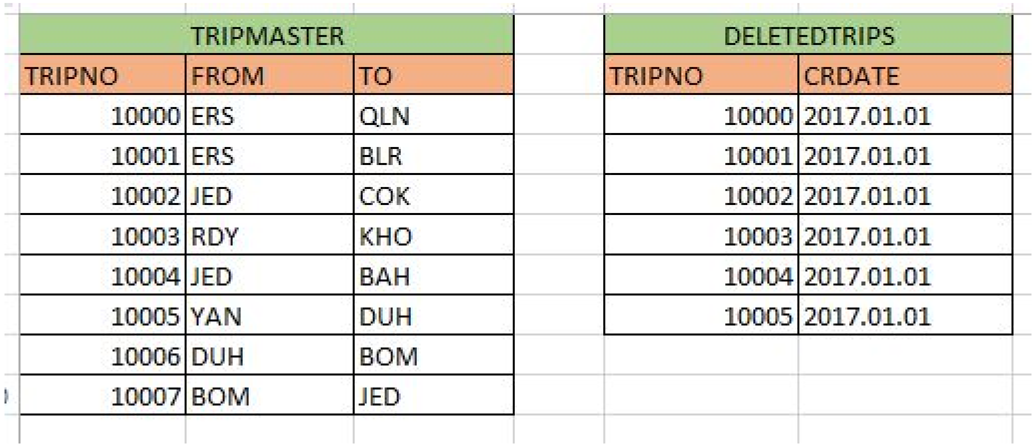
Now, the outer select fetches only records having CNT = 2. So , we got the records as required .

## 4. Logic Block – Entries found in one table and not in other

Title is little confusing; I couldn't find any better title for this scenario.

Wee have two tables  TABLEA and TABLEB. I want to fetch entries from TABLEA , which are not available in TABLEB.

In the below example, the TRIPMASTER table has all the trips created in the system. When a trip is deleted , DELETEDTRIPS table is updated. Requirement here, is to get the trip details which are not deleted. Technically, select entries from "TRIPMASTER" whos trip numbers are not present in DELETEDTRIPS.



| TRIPMASTER | | |
|---|---|---|
| TRIPNO | FROM | TO |
| 10000 | ERS | QLN |
| 10001 | ERS | BLR |
| 10002 | JED | COK |
| 10003 | RDY | KHO |
| 10004 | JED | BAH |
| 10005 | YAN | DUH |
| 10006 | DUH | BOM |
| 10007 | BOM | JED |

| DELETEDTRIPS | |
|---|---|
| TRIPNO | CRDATE |
| 10000 | 2017.01.01 |
| 10001 | 2017.01.01 |
| 10002 | 2017.01.01 |
| 10003 | 2017.01.01 |
| 10004 | 2017.01.01 |
| 10005 | 2017.01.01 |

An inner join would have sufficed if the requirement was to fetch records found in both the tables. We have many methods to run this logic in ABAP layer; like fetch records, loop the first table, then read from second table and so on.

Let's try to achieve the same using SQL SCRIPT.

Lets use a left outer join.

When there is no match in the right table for the given join condition, the projection from right table will be null. Which means; If I Left outer join TRIPMASTER and DELETED TRIPS, All the trips which are not found in the right table(DELETEDTRIPS) shall be null. using an wrapper SQL, we can filter the records where field is null.

```
 SELECT TRIPNO , FROM , TO from
(
SELECT TM.TRIPNO, TM.FROm, TM.TO , DT.CRDATE from
TRIPMASTER TM
left outer join
DELETEDTRIPS DT
on TM.TRIPNO = DT.TRIPNO
where ...
) where CREDATE is null;
```

CRDATE from **DETEDTRIPS** table will be projected as null, if there is no matching in

## What's in part 2

- How to push selection screen data effectively to HANA layer / CDS

- Analyze pain ABAP logic blocks
- When to choose CDS , Table Function & AMDP logically
- Design thinking -ABAP HANA for Web applications.
- And much more,.

Read my other related blogs.

AMDP – What's happening

Title credit : Michelle Crapo

*You see this message because you read the whole blog.  care to leave a comment and like the blog?*

Sreehari V Pillai

SAP HANA | ABAP Development | Measure Software Performance | abaponhana | codepushdown |

## Related Blog Posts

SAP HANA Cockpit Performance Comparison APP Use Case
By **Hai Zhang** , Dec 11, 2019

Three ways to check the push-down of filters
By **Jan Zwickel** , Aug 25, 2017

Flags to enforce the push-down of filters (available SAP Web IDE since SAP HANA 2.0 SPS02)
By **Jan Zwickel** , Aug 23, 2017

## Related Questions

Oracle to HANA database migration.. Things should be followed by ABAPer
By **ABC XYZ** , Dec 18, 2019

How to use BAPI/FM in HANA
By **Former Member** , May 24, 2014

HANA Housekeeping
By **Srikar Vankadaru** , Apr 24, 2014

## 12 Comments

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

Raja Dhandapani

March 19, 2018 at 9:08 am

Good Illustration!

Thank you!

Like (1)

---

Jelena Perfiljeva

March 19, 2018 at 9:03 pm

Quite confused by this blog… The title is a bit misleading. ABAP is still ABAP and the term “mapping” is usually used when it comes to data. (Translation issue?)

It says “for beginners” but then “We all already know, what is CDS and AMDP”. Well, if we’re beginners then maybe we don’t? Why not just start with a short explanation?

Item 1: why is this a table function? How is it used?

Item 2: why is it using MSEG and not the new HANA table MATDOCU?

Item 3: I think there is an FM for that but in any case even in the conventional ABAP you probably don’t need two separate SELECTs. And “inactive” flag in JEST is not considered? That’s odd…

Item 4. Not really sure what it’s telling us. LEFT JOIN existed in ABAP for a very long time.

The story started with updating an old program. Seems like a valid case but then where is the “before” and “after” code. That would be more beneficial IMHO than what looks like a random collection. I feel you could do much better than this.

Thank you.

Like (1)

Sreehari V Pillai | Post author

March 20, 2018 at 10:44 am

Jelena ,
Thanks for your feedback. I agree , that the title could have been more meaningful and less confusing .

ABAP is not still ABAP any more.

There are huge number of ABAPers who are yet to get exposure to practical implementation and use of ABAP on HANA features . This blog targets audience those just know new features part of ABAP on HANA but struggle to make use of it in practical scenarios.
There are plenty of other blogs and tutorials about what's CDS, AMDP and Table function . This blog does not intent to describe it.

Item 1: As I said , my intention is not to explain Table function – There a lot available in the community

Item 2 – Its a common misunderstanding. S4H and Suite on HANA are not the same. MATDOC is available only in S4H and not in Suite on HANA. I chose a scenario to explain how can we write it using SQL Script. May be I could have used the cliché SFLIGHT scenario which everyone is fed up with.

Item 3 : Function module ? My topic of interest is code pushdown to DB layer . Not to keep the logic in Application layer. In reality, I use inactive flag in JEST to reduce the project records. As I said before, functionality is not hat I am focusing. Also , are you in a mood to accept a challenge ? write it in a single SELECT using conventional ABAP(I tried and I couldn't) – Will be a great reference to us.

Item 4 : This is just an idea . I haven't seen such a logic written else where( even in SAP's standard programs)- I mean, using a LOJ to achieve the described scenario.

its not about updating an old program – Its about design thinking. There are many people who still write their code using old programming paradigm. Its for them.

Love – Sreehari

Like (1)

Young Hwan Kim

March 20, 2018 at 4:38 am

Useful. Something that I haven't thought before. Looking forward the next one.

Like (1)

---

Joachim Rees

March 21, 2018 at 12:59 pm

I – like Jelena – was a little confused by the title, but I think you are giving quit a few nice examples here!

Thanks a lot!

Joachim

Like (1)

Sreehari V Pillai | Post author

March 21, 2018 at 3:30 pm

Thanks man. Any suggestion on a less confusing title ? 😬

Sree

Like (0)

---

Michelle Crapo

March 22, 2018 at 11:44 am

This is a great blog! It shows the different examples that make sense.

1. Title – what's a title really? It's the first thing a person sees to determine if they should read the blog. Yes, your title is confusing.  How about something like: "Horrible Performance, HANA push down to the rescue", "Tips and Tricks using Push Down Technology", or "Side by side examples classic ABAP to CDS"
2. So where did the title go wrong?  Empower would probably work. "your ABAP with HANA". It is with HANA but it is more the languages being used. "Conventional ABAP to ABAP HANA" I really thing this works against you.  I see that and would expect perhaps Use push-down ABAP techniques to solve performance issues.

I like the way it was put together. You could have put your entire program and then the new program. That wouldn't be real useful. But what you did is leave us with some Tips and Tricks to make our development easier.  Because this is a blog about performance – I would look at the best way to do things. If it doesn't happen to be the best way, that's OK, someone will comment.

Long story – short or short story long – I really liked this. It will be one of my bookmarks. There are some solid ideas presented with how to fix.

Thank you for this new addition to some of my tips and tricks I keep with me!

Michelle

Like (0)

Sreehari V Pillai | Post author

March 22, 2018 at 12:06 pm

This comment made my day ! You must have seen the old title – Which was pretty confusing ! Thanks for the titles suggestions too .

Sree

Like (0)

Sreehari V Pillai | Post author

March 22, 2018 at 12:13 pm

And I really thought about explain end to end , how we revamped a conventional abap report to kick-ass Fiori + CDS report.

I am bit worried about the Intellectual property management ( customer policy ) here to publish any content outside . But, I am planning to write a blog in general soon for sure,.

Sree

Like (1)

Michelle Crapo

March 22, 2018 at 12:19 pm

Yes, customer policy drives what we can share and what we can't share.

I'm looking forward to the next blog,

Michelle

Like (0)

---

Chandan Saurav Panda

September 19, 2018 at 11:25 am

Hi Sreehari,

This is a nice blog. You have explained clearly how to handle few scenarios in SQL, rather than doing the same on ABAP.

I have just one suggestion. In the part 3, why can't we use a HAVING clause? I think the outer select is not required.

```
    lt_via_out = SELECT * from  (  select   v3.objnr as objnr ,
                                   count(*) as cnt
                                  FROM :lt_via v3 inner join jest j
                                 on v3.objnr = j.objnr
                                  AND J.stat IN ('I0001' , 'I0002' )
                                   group by v3.objnr )
  where cnt = 2;
```

Thanks,

Chandan

Like (1)

Sreehari V Pillai | Post author

October 15, 2018 at 4:06 am

Thanks Chandan .

Thats right. We can apply HAVING clause and avoid the nested SQL 🙂 thanks for pointing it out.

Sreehari

Like (0)