

# Modern ABAP Questions

This document only includes answers to key questions that were not discussed during the live course.

## Week 2: Developing Core Data Services (CDS) for ABAP

The answers to questions asked during Week 2 of Modern ABAP (*Developing Core Data Services (CDS) for ABAP*), held on September 17, 2024, are presented here. In some instances, multiple questions about the same topic have been combined (as denoted by the tag “cluster topic”).

### 1. Could you please clarify whether Rheinwerk Publishing will help for getting access for cloud server for practicing CDS/AMDP?

This course doesn't include system access. However, you can play around using an SAP Business Technology Platform trial account, which is free at this time.

### 2. Cluster topic: Views vs. view entities

We use “define view entities” because “define view” creates an old kind of CDS view that is obsolete on new SAP versions.

### 3. Cluster topic: Benefits of CDS

First of all, CDS views are a must for ABAP RESTful application programming model (RAP) development. You can't develop a managed RAP application with Open SQL. In terms of on-premise ERP development (SAP S/4HANA), CDS brings advantages like:

- Centralized queries, which are testable and reusable
- Enhanced syntax for powerful calculations
- Code pushdown for performance optimization (except in very complex cases; then, ABAP-managed database procedures [AMDP] are recommended)
- Nested queries between multiple CDS views
- Independent data APIs for complex table structures (data abstraction)
- Data units for AMDP/SQLScript development
- Built-in authorizations for increased security
- Annotations for additional metadata

### 4. Cluster topic: CDS performance

In most cases, the SAP HANA database engine is smart enough to prevent redundant queries. In simple cases, you can call CDS1 from CDS2 safely. In terms of nesting CDS views, I don't have a number

limit off the top of my head, but you can imagine it like a nested query targeting regular database tables. SAP HANA would act in the same way.

Performance problems may arise when you have aggregate functions or complex calculations in CDS1 and try to `JOIN` this table from CDS2. In other words, nesting complex CDS views can be problematic. The SAP HANA engine may not be able to calculate the most optimized path, or it may simply be forced to completely execute the child CDS before running the parent CDS—which may be slow on big tables. For such cases, AMDP/SQLScript features provide superior performance in many cases. That’s the topic of next week. (SQLScript can also be debugged easily, which is a further benefit for complex cases.)

My preference would be: 1) CDS in simple cases 2) CDS powered SQLScript (or plain SQLScript) in complex cases and 3) OpenSQL if I must.

## 5. Cluster topic: CDS and ABAP RESTful application programming model

The SEGW-based ABAP programming model for SAP Fiori and the CDS-based ABAP RAP model should generate a similar RESTful output—but details are different. In the old world, we would generate the backend service over SEGW and consume it in an SAP Fiori frontend application. In the new world, we generate the backend service using a RAP toolset, which is based on CDS views. The published service contains a RESTful API, as well as annotations to be consumed by the SAP Fiori elements frontend application.

Annotations declared on top of CDS views enable us to describe on the backend what the frontend should look like, and the SAP Fiori elements frontend application can act accordingly, building and displaying the frontend automatically. Then, we can modify the frontend as needed.

You can still develop freestyle frontend applications, but it is not needed in many cases.

You can’t use transaction codes from the old ABAP programming model for SAP Fiori to initiate RAP tests for virtual elements. Those are two separate mechanisms. Instead, you can create a quick RAP service and initiate a test directly from within Eclipse.

Compared to regular CDS columns, virtual elements will have an inevitable performance cost. Therefore, it should only be preferred when necessary.

Custom entities are typically used when your data retrieval involves complex logic, such as BAPI calls, Z-Class instances, custom error handling, or complex value calculations.

## 6. Cluster topic: [1..1] Associations

`VBAK` would have a [1..1] association to `KNA1` via `KUNNR`.

As in any association, nothing happens if you don’t access fields in the associated table. If you do access those fields and there is no correspondence in the [1..1] table, SAP HANA will still return the main entries, but leave the associated entry values as `NULL`. So, you can think of it like an outer join.

Example: This is my core CDS, which has a non-corresponding association on purpose.

```
define view entity ZTEST_INVALID_11 ASSO
as select from kna1
```

```

association [1..1] to mara on mara.matnr = knal.kunnr
{
  key kunnr,
    mara
}

```

Here is my consumer CDS:

```

define view entity ZTEST_INVALID_11 ASSO_CONS
as select from ZTEST_INVALID_11 ASSO
{
  kunnr,
  mara.meins
}

```

When I query the consumer CDS, I will still see `KUNNR` values, but `MARA.MEINS` column will be `NULL`.

## 7. With all the benefits of associations, should I always use them instead of regular CDS views?

Depends on your purpose. If you are targeting fields, which must exist in your CDS, I would still use INNER or OUTER JOINS. INNER JOINS are also useful to assert connections between tables. For optional satellite tables, I would prefer associations.

That being said, there is no harm putting associations into your CDS. Even if they are rarely used, they act like technical documentations regarding your table structure.

If your table relationships are needed in the future, the responsible developers can easily utilize that over associations—reaping the benefit of code centralization. Instead of repeating the same JOIN conditions over and over again, your centrally defined association conditions would be reused.

If your association logic changes over time due to a table structure change, it would be enough to change association definitions only—without changing a single line of code in client applications.

## 8. Cluster topic: CDS parameters

CDS views can have default parameters—to a degree, yes. Check the following code:

```

define view entity ztest_def_param
with parameters
  @Environment.systemField: #SYSTEM_DATE
  erdat : abap.dats
as select from vbak
{
  key vbak.vbeln
}

```

```
where
    erdat = $parameters.erdat
```

We can consume this CDS in classic ABAP without passing a value to ERDAT. The default system date will be used.

```
select * from ztest_def_param into table @data(vbaks).
```

You are limited to a handful of system fields to set as default values though.

```
@#CLIENT (annotation)
@#SYSTEM_DATE (annotation)
@#SYSTEM_LANGUAGE (annotation)
@#SYSTEM_TIME (annotation)
@#USER (annotation)
@#USER_DATE (annotation)
@#USER_TIMEZONE (annotation)
```

One common workaround for default parameter values is to create a second wrapper CDS, which passes default values to the child CDS.

The benefit of having an import parameter is the same as any subroutine. You can enforce the client to provide values (such as date, company code, user, etc.), which would change the output of the CDS. But the most significant use case would be table functions (which is the topic of next week).

#### 9. Is the \$session. fields only available within CDS views or also within classic ABAP code?

This is a CDS-only variable. In classic ABAP, you have SY for that. In SmartForms, you have SFSY.

#### 10. Can I consume CDS with parameters in another CDS view.

Yes, we did that during the course. Check the code samples.

#### 11. You can't use parameters on navigation with associations/based intent navigation, right? For a better performance when you have multiple pages (list report page + object pages). The only thing for optimization is to create many intermediate CDSs, not have a big CDS with many joins, right?

I agree with your approach. I would avoid parameters on CDS view entities targeting RAP. And instead of building big CDS entities, I would create multiple CDS entities and bind them with associations.

“God object” is an anti-pattern, and a typical code smell.

#### 12. Cluster topic: CDS debugging and monitoring

I'm not aware of a feature that enables us to debug a CDS view directly. Which makes sense too—when debugging ABAP code, you can F6 through an Open SQL statement, but you can't enter the statement because it's ultimately executed in the database kernel. The same applies to CDS.

However, if you split your CDS into multiple CDS views via associations (as you should), you can use the Eclipse preview functionality to drill down your views through associations and find problematic cases.

But you can certainly debug AMDP/SQLScript code! That is included in our session next week, and it's one of the reasons to prefer SQLScript in complex cases.

In ERP systems, you can use your regular performance monitoring tools. If your program is spending too much time on a CDS, it will be shown like any other table, and you can inspect the CDS closer for performance improvement.

### 13. Are SAP HANA XS and SQLScript the same thing?

No. SAP HANA XS is an application server built into SAP HANA, which can host web-based applications directly on the database. SQLScript is a programming language to write stored procedures within the SAP HANA environment, similar to TSQL, PL/SQL, PL/pgSQL, etc.

### 14. Would I be able to use that calculated bonus amt in WHERE clause?

If you calculate a bonus amount in CDS1, you can use it in the where clause of CDS2 which reads data from CDS1. Be careful though—if CDS1 is accessing a large table and run aggregations / functions, you may experience serious performance problems. AMDP/SQLScript should be preferred for such complex cases, which is the topic of next week.

### 15. Where do CDS views take rates from? Exchange rate type?

In SAP ERP, the system would be configured to read exchange rates from TCURR and its satellite tables. For SAP BTP, I recommend taking a look at <https://help.sap.com/docs/btp/sap-business-technology-platform/currency-conversion>.

### 16. Is it possible to cascade string functions?

You can use nested string functions, as well as other CDS functions.

### 17. Can we use RegEx in CDS views?


To an extent, yes. Check the command REPLACE\_REGEXPR, as well as LIKE\_REGEXPR and OCCURRENCES\_REGEXPR.

### 18. Cluster topic: CDS extensions and authorizations

You can extend any CDS that has the appropriate annotation on top—like `#PROJECTION_LIST`.

You can find extensions or authorization objects simply via a Where Used List.


References for: ZTEST\_CDS\_01 (Data Definition) [TDE] - ? matches in 2 objects

>  ZTEST\_CDS\_01\_DA01 (Access Control) - ? matches

>  ZTEST\_CDS\_02 (Data Definition) - ? matches

A CDS view can have multiple data access controls, as displayed below.

References for: ZTEST\_CDS\_01 (Data Definition) [TDE] - ? matches in 3 objects

>  ZTEST\_CDS\_01\_DA01 (Access Control) - ? matches

>  ZTEST\_CDS\_01\_DA02 (Access Control) - ? matches

Here is a case study: This is what my CDS results look like initially, before any kind of access control.

RB reqid	RB apord	RB seqnr
0000000055	001	00001
0000000057	001	00001

My first access control, limiting results to reqid = 55:

```
@EndUserText.label: 'Data access 01'
@MappingRole: true
define role ZTEST_CDS_01_DA01 {
  grant
    select
      on
        ZTEST_CDS_01
      where
        reqid = '0000000055';
}
```

Results after that:

AB reqid	AB apord	AB seqnr
0000000055	001	00001

My second access control, limiting results to reqid = 99:

```
@EndUserText.label: 'Data access 02'
@MappingRole: true
define role ZTEST_CDS_01_DA02 {
  grant
  select
  on
    ZTEST_CDS_01
  where
    reqid = '0000000099';
}
```

Results after that, still the same:

AB reqid	AB apord	AB seqnr
0000000055	001	00001

So, multiple access controls will run “optimistically”—for lack of a better term.

### 19. For literal values or constants can we reference type group or public static constants from a class? Having IT literally embedded into CDS is difficult from a maintenance aspect.

You can put your static values into a domain and read tables DD01L / DD07L / DD07T from your CDS. Or simply create a Z table containing your constants, and access that table from your CDS.