

Technical Articles



Andrea Schlotthauer

October 16, 2021 | 6 minute read

A new generation of CDS views: how to migrate your CDS views to CDS view entities

Follow



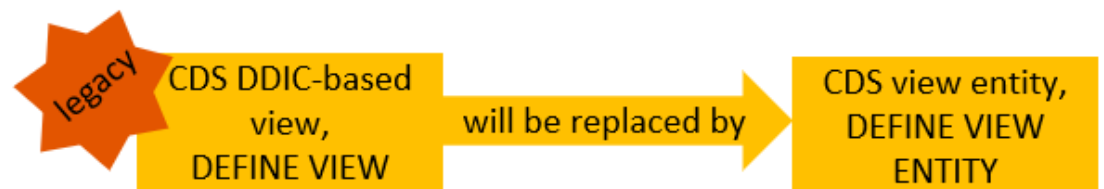
Like



RSS Feed

2 14 1,966

With ABAP release 7.55, a new type of CDS view has been released: the **CDS view entity**. CDS view entities are an improvement over the “classic” CDS DDIC-based views.



For details, see the [blog post about CDS view entities](#).

New migration tool available

Now, a new **migration tool** is available that automates many steps of a migration from a CDS DDIC-based view to a CDS view entity. But that’s not the only helper to facilitate a migration. The following support has been released:

What	Technical name	Release
Program for migration analysis	RUTDDL_S_MIGRATION_CANDIDATES	ABAP Release 7.55 ABAP Platform 2020
Manual migration		ABAP Release 7.56 ABAP Platform 2021
Program for tool-based migration	RUTDDLSV2MIGRATION	ABAP Release 7.56 ABAP Platform 2021

This blog post first provides some general information about migration and then gives details on each helper tool.

General considerations

View entities are by design incompatible with DDIC-based views.

The advantages of CDS view entities can only be realized based on incompatible changes to the concepts and implementation of existing CDS views. The most prominent example is the deletion of the CDS-managed DDIC view, that means, the ABAP Dictionary SQL view that is attached to each DDIC-based view. This DDIC SQL view is deleted during a migration to a CDS view entity and all scenarios which rely on the DDIC SQL view no longer work.

No automatic migration will take place.

During migration, semantics of a view might change. Because of this, each developer needs to decide whether they want to migrate their views.

Reverse migration

To reverse-migrate from view entity back to DDIC-based view just open a support ticket on BC-DWB-DIC.




Program for migration analysis

Since ABAP On-Premise Release 7.55, an ABAP program is available that evaluates whether a migration from a CDS DDIC-based view to a CDS view entity is possible:

RUTDDL_S_MIGRATION_CANDIDATES

You can select DDLS sources to be analyzed directly by name or indirectly by original system, person responsible, package, and so on.

As a result, all DDLS sources specified are classified into one of the following categories (column *Overall Status*):

 Red	Migration is not possible due to a technical constraint.
 Yellow	Migration is possible with some potential behavior changes. Changes in the DDLS source are required.
 Green	Migration is possible, though some minor changes might be required.

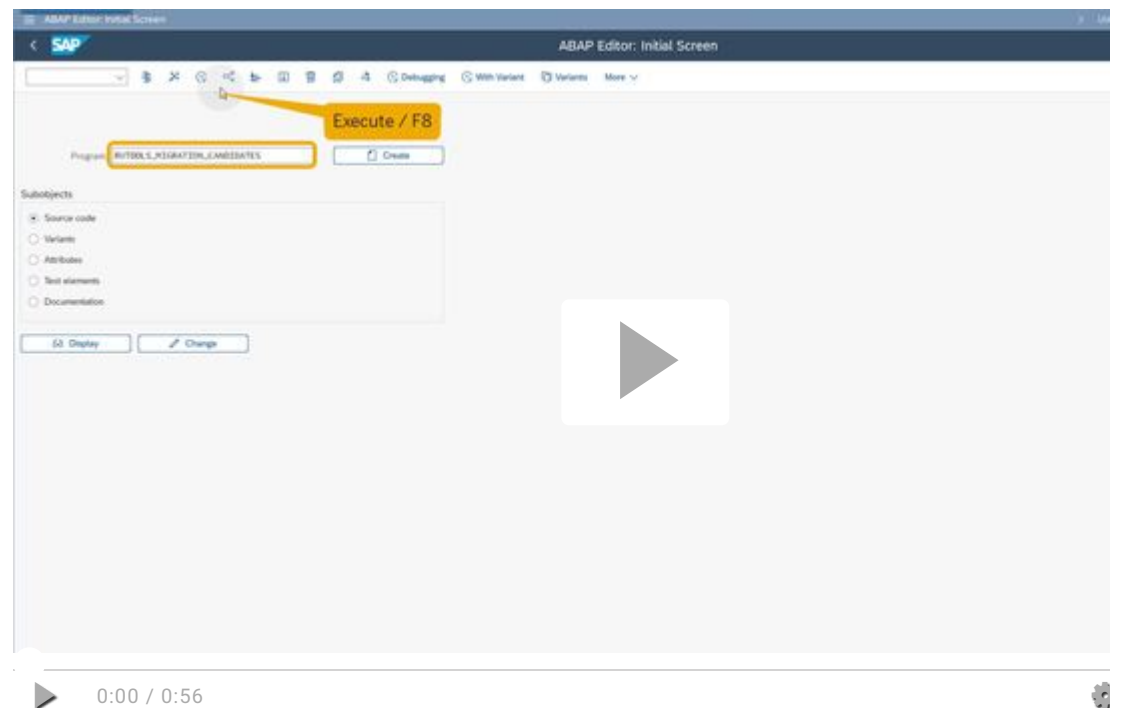
Moreover, details on the sources of the errors are provided. Currently, the following aspects are checked for each migration candidate:

- Entity and DDLS name of the migration candidate must be identical. Otherwise, migration is not possible.
- The decimal shift function `DECIMAL_SHIFT` cannot be used in a migration candidate, since it is not supported in CDS view entities. CDS view entities offer the following functions instead:
 - `CURR_TO_DECFLOAT_AMOUNT`
 - `GET_NUMERIC_VALUE`
- The data source of the migration candidate cannot be a DDIC external view, since DDIC external views are not allowed in CDS view entities.
- Buffering annotations cannot be used, since they are not supported in CDS view entities.
- Certain obsolete data types are not supported in CDS view entities. If the migration candidate uses one of the obsolete types, you need to change this before a migration is possible.
- `SELECT *` is not supported in CDS view entities. You need to formulate an explicit element list instead.
- The annotation `@AbapCatalog.compiler.compareFilter` is not supported in CDS view entities. In CDS view entities, this annotation is always implicitly set to `true` and this default can't be changed. During a migration, you need to remove this annotation and this might result in a behavior change:
 - If the migration candidate uses this annotation with the value `false` and you remove it as part of your migration, then the value is implicitly set to `true`.
 - If the migration candidate doesn't use this annotation at all, then the implicit default value is `false` and during a migration, the implicit default behavior is changed from `false` to `true`.
- The annotation `@ClientDependent` is not supported in CDS view entities. Client handling is done automatically and implicitly in a CDS view entity by filtering the session variable `$session.client`. You need to remove the annotation `@ClientDependent` during a migration and this might result in a behavior change.
- The syntax `$EXTENSION.*` is not supported in CDS view entities.
- The DDIC SQL view of the migration candidate is used by other objects or ABAP programs. These dependent objects will no longer work after a migration.
- If the migration candidate includes input help, the direct assignment is lost during a migration.

For more information on an error, including how to solve it, select an error category and click F1.

CDS Entities											
DDL Source	Entity Name	View name	Overall Status	DDLs STOB	Dec Shift	Ext View	Buffering	Data Types	Select * Check	ComFilt()	AnoClicDe
ANDREA_AR...	ANDREA_CD...	HIREEGHIR...	●○○	●○○	○●●	○○●	●○○	○○●	○○●	○○●	○○●
ANDREA_MA...	ANDREA_MA...	BGGREG	●○○	○○●	○●●	○○●	●○○	○○●	○▲○	○○●	○○●
DEMO_CDS...	DEMO_CDS...	COALV1	○▲○	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
DEMO_CDS...	DEMO_CDS...	DEMOCDSD...	○▲○	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
ANDREA_TE...	DEMO_CDS...	DEMO_CS...	●○○	●○○	○○●	○○●	○○●	○○●	○○●	○▲○	○○●
DATE_FUNC...	DEMO_CDS...	DEMO_CDS...	●○○	●○○	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
FLTP_TO_D...	DEMO_CDS...	DEMO_CDS...	●○○	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
DEMO_SALE...	DEMO_SALE...	DEMO_SALBP	○▲○	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
DEMO_SALE...	DEMO_SALE...	DEMO_SALE...	○○●	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
DEMO_SALE...	DEMO_SALE...	DEMO_SAL...	○○●	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
DEMO_SALE...	DEMO_SALE...	DEMO_CDS...	○○●	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
DEMO_SALE...	DEMO_SALE...	DEMO_SALE...	○○●	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
DEMO_SALE...	DEMO_SALE...	DEMO_SAL...	○○●	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
DEMO_SALE...	DEMO_SALE...	DEMOANALY...	○▲○	○○●	○●●	○○●	○○●	○○●	○○●	○▲○	○○●
ZZAS_MIGV2...	ZZAS_MIGV2...	ZZASMIGV2...	○○●	○○●	○●●	○○●	○○●	○○●	○○●	○○●	○○●
ZZAS_MIGV2...	ZZAS_MIGV2...	ZZAS_MIGV2...	○○●	○○●	○●●	○○●	○○●	○○●	○○●	○○●	○○●
ZZAS_MIGV2...	ZZAS_MIGV2...	ZZASMIGV2...	○○●	○○●	○●●	○○●	○○●	○○●	○○●	○○●	○○●

Video clip that demonstrates the migration analysis tool:



Manual migration support

Since ABAP 7.56, ABAP Platform 2021, manual migration from a CDS DDIC-based view to a CDS view entity is supported. To perform a manual migration, follow these steps:

1. Check whether a migration is possible using the migration analysis report RUTDDL_S_MIGRATION_CANDIDATES.
2. Open the DDLS source in ADT and perform the following steps:
 - i. Remove the header annotation *sql/ViewName*.
 - ii. Add the keyword *ENTITY*.
 - iii. Perform additional changes if required. The migration analysis report and the syntax check provide feedback on the required adjustments.
3. Activate the view entity.

Tool-based migration

A migration tool is available since ABAP 7.56, ABAP Platform 2021:

RUTDDLSV2MIGRATION

The migration tool automatically performs many adjustments that are required for a migration and creates an inactive version of the migrated DDLS. It also includes many checks and in case of issues, indicates which lines of the migration candidate must be changed before a migration is possible. The last step of the migration – the final activation of the CDS view entity – must be done by the developer.

When using the migration tool, follow these steps:

1. Start the ABAP program RUTDDLSV2MIGRATION.
2. Select the DDLS sources to be migrated. Objects can be selected directly by name or indirectly by package, original system, person responsible, and so on.
3. Choose the execution mode. We recommend that you first perform a migration in simulation mode and afterwards execute the actual generation of an inactive view entity.
4. Choose *Execute*.

The migration consists of four phases:

- i. Initial Consistency
- ii. Precheck

Note: To see the embedded help, double-click on one of the prechecks. The same checks are performed as with the migration analysis report RUTDDL_S_MIGRATION_CANDIDATES. See the explanation above.
- iii. Rule Processing
- iv. Post Check

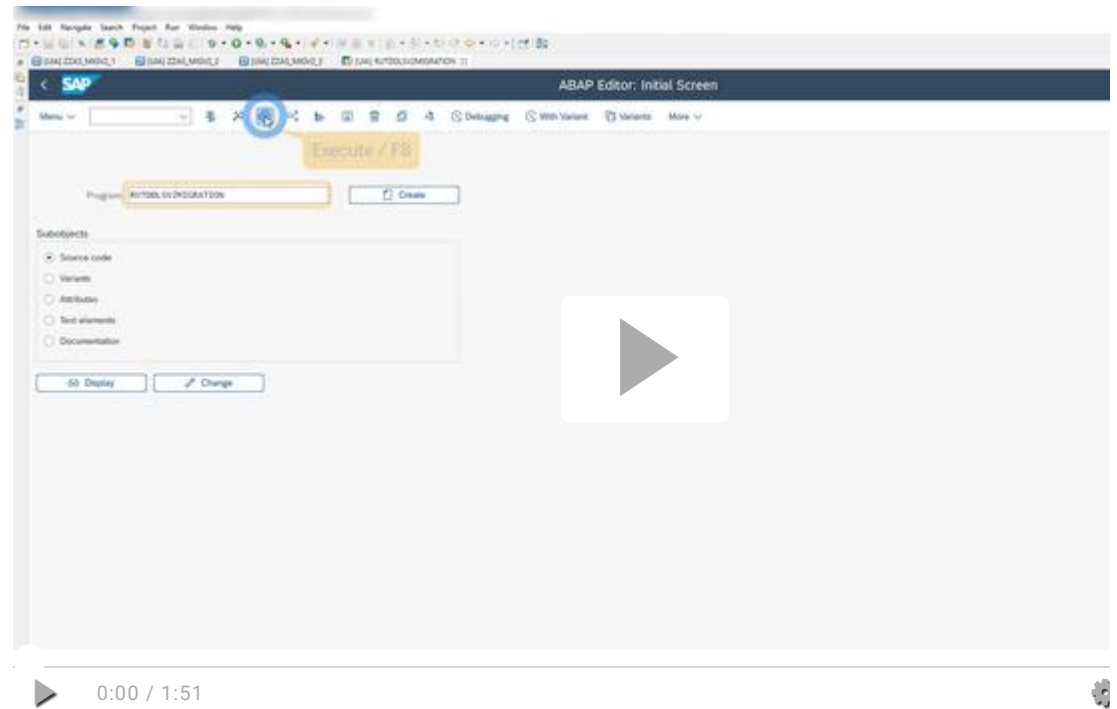
The tool provides an overview of the differences between the CDS view before the migration and the CDS view entity after the migration. It also gives feedback on each phase and indicates errors that must be solved manually (if applicable).

5. Activate the view entity.

The following list explains the rules that are applied to a CDS view to turn it into a CDS view entity in chronological order:

- ANNOTATIONS_HEADER_PURIFY: Remove unsupported header annotations.
- SELECT_EXPAND_ASTERISK: Replace SELECT * by explicit data source field list.
- MOVE_SELLIST: Move the selection list behind the FROM clause and embrace it with { }.
- ANNOTATIONS_REFERENCE_ADD: Add *@Semantics.amount.currencyCode* and *@Semantics.quantity.unitOfMeasure* if required.
- ANNOTATIONS_REFERENCE_CLEAR: Remove *@Semantics.currencyCode* and *@Semantics.unitOfMeasure*.
- LITERAL_BACKSLASH: Add a backslash (\) to each backslash in CL_QLAST_LITERAL.
- PARAMETER_SYNTAX_CORRECT: Change parameter syntax from *:ParamName* to *\$parameters.ParamName*.
- FUNC_CEIL_FLOOR: Remove superfluous function call for integral parameters.
- NAMELIST_REMOVAL: Remove name list (signature).
- FUNC_AVG_ADD_TYPE: Add mandatory type to function AVG().
- BASEOBJ_SQLVIEW_TO_STOB: Change SQLVIEW names to STOB names of migration candidate data source.
- ELEMENT_PREFIX: Add element (associations, fields) prefixes if required.
- ALIAS_ADD_AS: Add keyword AS to aliases.
- ANNOTATION_ANALYTICS_TECHNICAL_NAME: Add the annotation *@Analytics.technicalName: 'SQLViewName'* if required.
- ASSOC_ON_ELEMENT_NAMES_CORRECT: Correct element names in association on conditions: Use expression *\$projection. ElemName* instead of *ElemName*.
- ANNOTATIONS_ARRAY_SYNTAX_CORRECT: Correct syntactically wrong array annotations.
- DOMAIN_FIX_VALUE: Replace domain fix value with literal value.
- UNION_ADD_ANNO_PREVENT_INHERITENCE: Generate the following mandatory annotation in UNION views:
@Metadata.ignorePropagatedAnnotations: true.
- UNION_ALIGN_KEY_DEFINITION: Align key definition in all UNION branches to match the first branch.
- FIRST_ELEMENT_AS_CLIENT: Delete first field if it is a client key field.
- REMOVE_CLIENT_FROM_UNIT_CURRENCY_CONVERION: Remove explicit client column from UNIT_CONVERSION and from CURRENCY_CONVERSION.
- UNION_REMOVE_ANNOS_IN_2ND_PLUS_BRANCH: Delete all annotations from all the UNION branches except of the first one.
- ENTITY_EYECATCHER: Add ENTITY keyword to the DEFINE VIEW syntax.

Video clip that demonstrates the migration tool:



Conclusion and further information

Keep in mind: Migration is voluntary and “classic” CDS DDIC-based views are still supported, but not further developed.

You are welcome to leave comments and ask questions right here.

For more information, see the embedded help:

- RUTDDLSV2MIGRATION > *More* > *Program Documentation*
- RUTDDLS_MIGRATION_CANDIDATES > *More* > *Program Documentation*.

Assigned tags

ABAP Development

ABAP RESTful Application Programming Model

abap cds

abap cds views

migration tool

Similar Blog Posts



[CDS view entities - The new CDS views](#)

By Harish Bokkasam Sep 04, 2020

[Getting Started with ABAP Core Data Services \(CDS\)](#)

By Carine Tchoutouo Djomo Feb 01, 2016

[A new generation of CDS views: CDS view entities](#)

By Andrea Schlotthauer Sep 02, 2020

Related Questions



[CDS views with ABAP : 'The Parameter is not bound' error while activating ABAP program .](#)

By Sharda P Oct 04, 2021

[CDS Views with and without parameters - performance difference](#)

By Ab ABAP Oct 06, 2019

[Best practice for using CDS Views in structures](#)

By David Henn Sep 18, 2020

Join the Conversation



[SAP TechEd](#)

Tune in for tech talk. Stay for inspiration. Upskill your future.

[Coffee Corner](#)

Join the new Coffee Corner Discussion Group.

2 Comments

You must be [Logged on](#) to comment or reply to a post.

AjeethKumar R

October 17, 2021 at 7:56 am

Excellent blog, Andrea!!. Thanks for bringing this up.

Like 1 | Share

Christian Guenter

October 18, 2021 at 7:04 am

Nice.

Since ABAP On-Premise Release 7.55, an ABAP program is available that evaluates whether a migration from a CDS DDIC-based view to a CDS view entity is possible:

Unfortunately RUTDDL_MIGRATION_CANDIDATES is not available in my 7.55 SP01 On-Premise system. Is it shipped with SP02? Is there a note for implementation?

Best regards

Christian

Like 2 | Share

Find us on

Privacy	Terms of Use
Legal Disclosure	Copyright
Trademark	Cookie Preferences
Newsletter	Support