ABAP 7.40 SP5/SP8 Releaseabhängige Änderungen

Webinar, 29.1.2016

cadaxo

Danke, dass ihr dabei seid!

Wir starten in ein paar Minuten

Johann Fößleitner
Senior Consultant

Ich bin ABAP

SAP Entwickler seit 25 Jahren

Domik Bigl
Senior Consultant

Geht nicht gibt's nicht

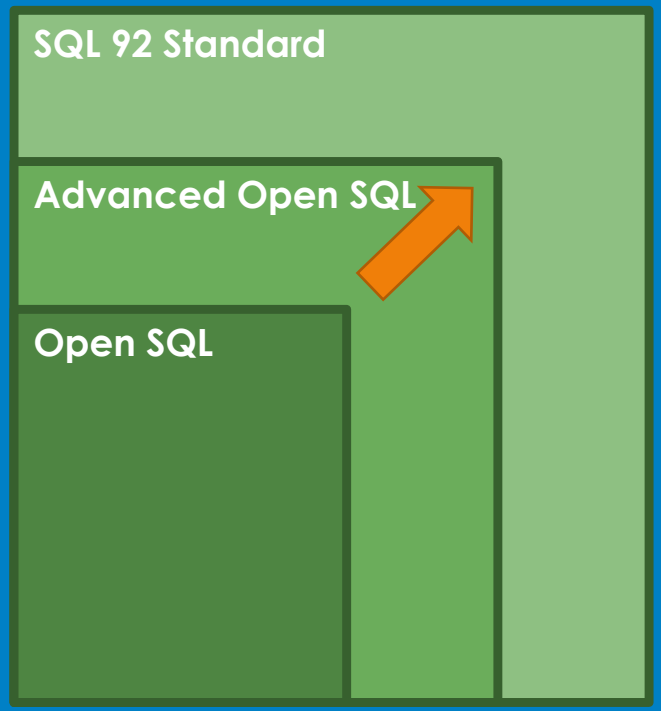
Unterwegs in SAP seit 2001

cadaxo

- ABAP New Open SQL
  - Neue Syntax, SQL Expressions
- ABAP Core Data Services
  - CDS Views
- ABAP Messaging Channels
- ABAP Push Channels
- Ausdrücke und Funktionen
- Sonstiges

Agenda

- **SQL Einschränkungen in ABAP vor 7.40**
  - Jahrelanger Stillstand
  - Kein UNION bzw. UNION ALL
  - Einschränkungen betr. JOIN Typen
  - Keine Expressions
  - …



Open SQL vor 7.40

- **Code Pushdown to the Database**
  - ABAP Open SQL Expressions
  - ABAP Core Data Services (CDS Views)
  - ABAP Managed Database Procedures (AMDP)

- **Neue Open SQL Syntax**
  - Escaping von Hostvariablen mit @
  - Separierung von Elemente mit Kommas

```abap
DATA: L_HUGO TYPE C LENGTH 10.

SELECT mc_name1,
       @l_hugo    AS hugo,
       @sy-uname  AS userid,
       'X'        AS true
       FROM TABLE BUT000
       INTO @ls_result.
```

ABAP Code

Open SQL Expressions

- SQL Expressions – Beispiel CASE

```
* Case Expression
select mc_name1,
       case when type = '1' then 'Person'
            when type = '2' then 'Unternehmen'
            else 'Unbekannt'
       end as type_desc
       from but000
       …
```

ABAP Code

- **SQL Expressions**
  - Fallunterscheidung mit CASE
  - Arithmetic Expressions
    - +, -, *, /, ABS, CEIL, FLOOR, DIV, MOD, …
  - Verketten von Spalten mit &&
  - Typanpassung mit CAST für FLTP Dictionary Typen
  - COALESCE-Funktion
  - Festwerte

- **SQL Expressions können kombiniert werden**
  - Priorisierung mit Klammern

```
* Case Expression
select type,
       case when type = '1' then ( 'Person' && '/' && type )
            when type = '2' then ( 'Unternehmen' && '/' && type )
       end as type_desc,
       mc_name1
       from but000
       …
```

ABAP Code

CDS Views

cadaxo

- **CDS Core Data Services**
  - Next Generation Data/View Definition
  - CDS beinhaltet
    - Data Definition Language (DDL)
    - (ab 7.50 Data Control Language (DCL))
  - 100% in ABAP integriert
    - ABAP Dictionary
    - Transportmanagement
    - Modifikationsfrei erweiterbar
  - Pflege über ABAP in Eclipse
  - Zugriff via Open SQL

SE11 – traditioneller Datenbankview

- **Einschränkungen**
  - Keine Outer Joins
  - Keine komplexen Joins
  - Keine Kommentare
  - Kein UNION
  - Kein View -> View
  - …

```
@AbapCatalog.sqlViewName: 'ZCDSVIEW_DB'
/* Demo */
define view ZCDSVIEW_ENT
   as select from but000
   {
     partner,
     mc_name1,
     mc_name2,
   }
```

CDS View

- **Definition in DDL**
  - Syntax ist sehr ähnlich zu Open SQL
- **CDS View Entity: ZCDSVIEW_ENT**
- **CDS View DB: ZCDSVIEW_DB**

CDS Views

- **CDS Views**
  - Code Pushdown durch erweiterte View Funktionalitäten
    - Mehr Möglichkeiten als bei Open SQL Expressions
  - Volle Integration in die ABAP Welt (z.B. Transportmanagment)
  - Pflege (derzeit?) ausschließlich über ABAP in Eclipse
  - Open SQL Zugriff auf die Views natürlich möglich
  - CDS Views sind erweiterbar
  - View on View möglich

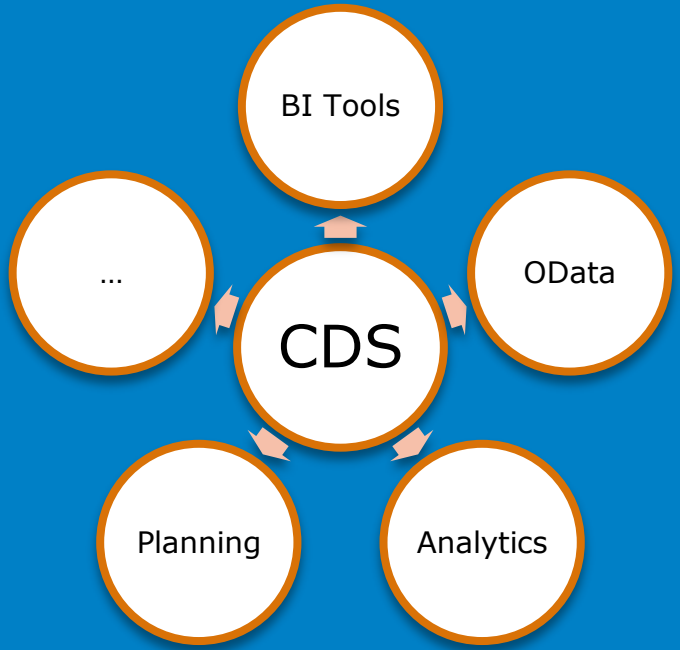
- **CDS Build-in Expressions**

Coalesce-Funktion	Konvertierungs-funktionen	Zeichenketten-funktionen	Numerische Funktionen	Zeit/Datum Funktionen
COALESCE	CURRENCY_CONV UNIT_CONV DECIMAL_SHIFT	CONCAT LPAD REPLACE SUBSTRING	ABS CEIL DIV DIVISION FLOOR MOD ROUND	DATS_IS_VALID TIMS_IS_VALID DATS_ADD_DAYS ...

Neu ab 7.50

Viele neue Funktionen ab 7.50

- **CDS Views - Annotations**
  - Ermöglicht das Anreichern von Metadaten direkt im View
  - Beispiele
    - SAP Pufferung aktiv
    - Mandantenhandling
    - Label bzw. Quickinfo
    - …



cadaxo

Core Annotations

Element Annotations

```
@AbapCatalog.sqlViewName: 'zcdsview_foe2_db'
@AbapCatalog.compiler.CompareFilter: true
@AccessControl.authorizationCheck: #CHECK
define view Zcdsview_Foe2
   with parameters exc_date: abap.dats,
                   to_curr: abap.cuky(5)
   as select from zdatentypen {
   zdatentypen.curr as original,
   @EndUserText.label: Currency'
   zdatentypen.cuky as currency,
   …
```

- **CDS Views Ausblick**
  - CDS Zugriffskontrolle ab 7.40 SP10
    - Der Zugriff auf CDS Views kann mit der neuen Datenkontrollsprache (DCL) geschützt werden
    - Verknüpfung mit PFCG Rollen möglich
  - Neue SQL Funktionen ab 7.50
    - CONCAT, LEFT, LENGTH, LTRIM, …
    - Eingebaute Datumsfunktionen (DATS_DAYS_BETWEEN, DATS_IS_VALID …)
  - Vertreterobjekte ab 7.50
    - Mit einigen Einschränkungen (und großer Vorsicht) können CDS Views als Vertreterobjekte für Datenbanktabellen definiert werden
  - …

- **Open SQL Ausblick ab 7.50**
  - **UNION/UNION ALL**
  - SQL Expression können auch in WHERE, HAVING, …
  - Neue SQL Expressions
    - CONCAT, LPAD, LENGTH, …
  - Subqueries als Source für INSERTS
  - **GTT** Global Temporary Tables
  - …

cadaxo

CDS Views

- **Re-Use Scenarien**
  - Views können verschachtelt werden!!
- **CDS Features**
  - ASSOCIATION, UNION, UNION ALL, Mengen bzw. Währungs Umrechnungen

Open SQL

- Einmalverwendungen
- FOR ALL ENTRIES

CDS Views vs. Open SQL Expressions

ABAP Channels

- **Bildschrimdaten aktualisieren bisher**
  - Useraktion (PAI oder Control-Events)
  - „Polling“ CL_GUI_TIMER
  - (JavaScript)

- **Datenaustausch Sessions/Benutzer bisher**
  - SAP-Memory (SET/GET Parameter)
  - ABAP-Memory (EXPORT/IMPORT)
  - Shared Objects
  - Datenbank

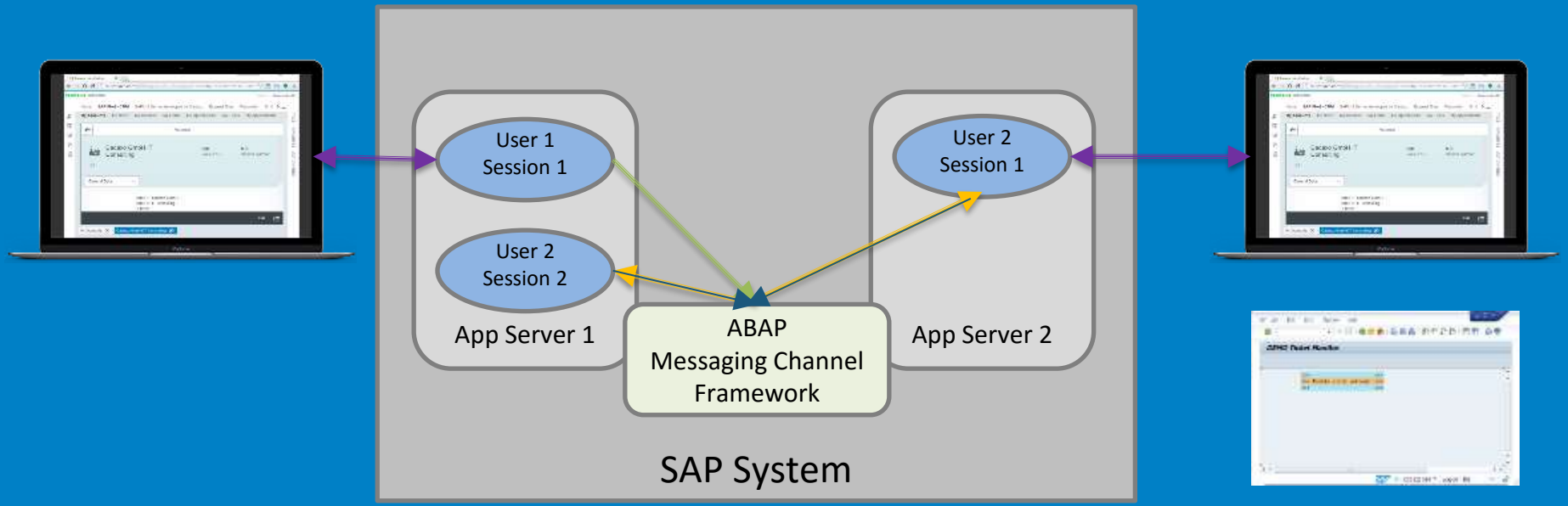
cadaxo

- **ABAP Push Channels - APC**
  - Kommunikation zwischen Client und Server
  - Bi-direktional: Server und Client
  - WebSocket Protocol

- **ABAP Message Channels - AMC**
  - Kommunikation zwischen Benutzersessions
  - Publish–Subscribe Model
  - AppServer übergreifend

ABAP Channels

APC Send/Receive
AMC Subscribe
AMC Send
AMC Receive



ABAP Channels - Überblick

cadaxo

- **LIVE DEMO - Szenario**
  - Benutzer erstellt Fehlerticket
  - Supportmitarbeiter erhält Ticket
  - Monitor im SAP GUI

- **LIVE DEMO**
  - Benutzer erstellt Fehlerticket
  - Supportmitarbeiter erhält Ticket
  - Monitor im SAP GUI

Ausdrücke, Funktionen und interne Tabellen

SP02
SP05
SP08

- **Konstruktorausdrücken / Inline-Deklarationen**
  - VALUE
  - NEW
  - CONV
  - COND/SWITCH
  - DATA()
  - FIELD-SYMBOL()

```
DATA(lt_but000)  = VALUE bup_but000_t( ). " TYPE STANDARD TABLE OF BUT000
DATA(ltr_but000) = NEW bup_but000_t( ).

"bisher
DATA lt_but000  TYPE bup_but000_t.
DATA ltr_but000 TYPE REF TO bup_but000_t.
```

Konstruktorausdrücken / Inline-Deklarationen

```
LOOP AT lt_but000 ASSIGNING FIELD-SYMBOL(<lwa_but000>).
  ...
ENDLOOP.
LOOP AT lt_but000 INTO DATA(lwa_but000).
  ...
ENDLOOP.

"bisher
DATA lwa_but000 LIKE LINE OF lt_but000.
LOOP AT lt_but000 INTO lwa_but000.
  ...
ENDLOOP.
```

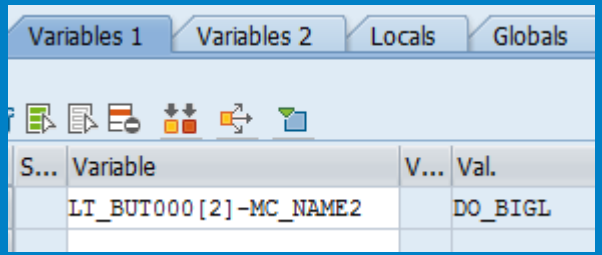
Konstruktorausdrücken / Inline-Deklarationen

- # Aufbau Rangetable

```
  "wfdst_partner_ranges_tab = TYPE RANGE TABLE FOR BU_PARTNER
  DATA(lrt_partner) = VALUE wfdst_partner_ranges_tab( sign   = 'I'
                                                      option = 'EQ'
                                                      ( low = 1 )
                                                      ( low = 2 )
                                                      ( low = 3 ) ).
*  "bisher
   DATA lrt_partner   TYPE wfdst_partner_ranges_tab.
  DATA: lrwa_partner LIKE LINE OF lrt_partner.

  lrwa_partner-sign   = 'I'.
  lrwa_partner-option = 'EQ'.
  lrwa_partner-low    = 1.
  APPEND lrwa_partner TO lrt_partner.
  lrwa_partner-low    = 2.
  APPEND lrwa_partner TO lrt_partner.
```

- **Tabellenausdrücke – „READ TABLE“**
  - ITab[2]
    - READ TABLE ITab INDEX 2 …
  - ITab[feld_a = `3`]
    - READ TABLE Itab WITH KEY feld_a = `3` …



- **Lesender Zugriff**

```
DATA(lt_but000) = VALUE bup_but000_t( ( partner = 1
                                       mc_name1 = 'DOMI'
                                       mc_name2 = 'BIGL' )
                                     ( partner = 2
                                       mc_name1 = 'JOHANN'
                                       mc_name2 = 'FOESSLEITNER' ) ).

DATA(l_fullname) = |{ lt_but000[ 2 ]-mc_name1 } { lt_but000[ 2 ]-mc_name2 }|.
" l_fullname = 'JOHANN FOESSLEITNER'

"bisher
FIELD-SYMBOLS: <lwa_but000> TYPE but000.
READ TABLE lt_but000 ASSIGNING <lwa_but000> INDEX 2.
CONCATENATE <lwa_but000>-mc_name1
            <lwa_but000>-mc_name2 INTO l_fullname
                                  SEPARATED BY space.
```

- **Schreibender Zugriff**

```
lt_but000[ 2 ]-mc_name2 = 'Föß'.
```

- **Parameterübergabe**

```
DATA(lr_msg_srv) = cl_bsp_wd_message_service=>get_instance( ).

IF lt_but000 IS NOT INITIAL.

  lr_msg_srv->add_message(
    EXPORTING
      iv_msg_type       = 'E'
      iv_msg_id         = '00'
      iv_msg_number     = '000'
      iv_msg_v1         = lt_but000[ 1 ]-partner ).

ENDIF.
```

Tabellenausdrücke

- **MOVE-CORRESPONDING ITabs**

```
DATA(lt_but000) = VALUE bup_but000_t( ( partner = 1
                                       mc_name1 = 'DOMI'
                                       mc_name2 = 'BIGL' )
                                     ( partner = 2
                                       mc_name1 = 'JOHANN'
                                       mc_name2 = 'FOESSLEITNER' ) ).
DATA lt_but020 TYPE TABLE OF but020.

MOVE-CORRESPONDING lt_but000 TO lt_but020.

"bisher
DATA: lwa_but020 TYPE but020.
FIELD-SYMBOLS: <lwa_but000> LIKE LINE OF lt_but000.
LOOP AT lt_but000 ASSIGNING <lwa_but000>.
  MOVE-CORRESPONDING <lwa_but000> TO lwa_but020.
  APPEND lwa_but020 TO lt_but020.
ENDLOOP.
```

MOVE-CORRESPONDING für interne Tabellen

- # Konstruktorausdruck: CORRESPONDING

```
DATA(lrt_partner) = VALUE wfdst_partner_ranges_tab( sign   = 'I'
                                                    option = 'EQ'
                                                    ( low = 1 )
                                                    ( low = 2 )
                                                    ( low = 3 ) ).

TYPES: ltt_but000 TYPE TABLE OF but000.
DATA lt_but000 TYPE ltt_but000.
lt_but000 = CORRESPONDING #( lrt_partner MAPPING partner = low ).

"bisher
DATA: lwa_but000 TYPE but000.
FIELD-SYMBOLS: <lrwa_partner> LIKE LINE OF lrt_partner.
LOOP AT lrt_partner ASSIGNING <lrwa_partner>.
  MOVE-CORRESPONDING <lrwa_partner> TO lwa_but000.
  lwa_but000-partner = <lrwa_partner>-low.
  APPEND lwa_but020 TO lt_but020.
ENDLOOP.
```

CORRESPONDING

- Weniger Hilfsvariablen
- Weniger LOOPs und READ TABLE
- Als Methodenparameter verwendbar
- Übersichtlicher
- Modernerer Programmierstil

Pro

- Zunächst weniger übersichtlich/verständlich
- Komplexer Aufbau
  - Datenobjekte
  - Methodenaufrufen
  - Debugger

Contra

- WHILE/UNTIL
- LET – Ausdrücke
- boolesche Funktion
- Tabellenfilterungen – Constructor FILTER
- Gruppierung interner Tabellen - FOR
- Constructor REDUCE
- Prädikative Methodenaufrufe
- OPTIONAL und DEFAULT bei Tableexpressions

Weitere Begriffe

- Klasse **cl_abap_corresponding**
- **IS INSTANCE OF** und **CASE TYPE OF**
- it_ref[3]->the_method( )

Ausblick 750

Sonstiges

- **Sonstige interessante ABAP Neuerungen in SP5/SP8**
  - **MOVE-CORRESPONDING** für interne Tabellen (SP5)
  - **WAIT UP TO** – Angabe ab nun in Millisekunden möglich (SP8)
  - **Outer Joins –** On-Condition nun auch <, <=, >, >=, <>, between und Literale verwendbar.
  - **Meshes**
    - Spezielle Strukturen die als Komponenten interne Tabellen haben welche über Assoziationen verknüpft sind.
    - Der Zugriff erfolgt über Mesh-Pfade
  - …

cadaxo

https://twitter.com/domibiglsap
https://www.linkedin.com/in/dominik-bigl-9b98b68b
https://www.xing.com/profile/dominik_bigl
dominik.bigl@cadaxo.com

Danke!

https://twitter.com/foessleitnerj
https://www.linkedin.com/in/johann-fößleitner-a9851b2a
https://www.xing.com/profile/johann_foessleitner
johann.foessleitner@cadaxo.com