

```

FUNCTION BAPI_INSPLOT_GETDETAIL
***Lokale Schnittstelle:
IMPORTING
  VALUE(NUMBER) LIKE BAPI
  VALUE(LANGUAGE) LIKE BA
  OPTIONAL
EXPORTING
  VALUE(INTERNAL_DATA) LIK
  STRUCTURE BAPI2
  VALUE(TASK_LIST_DATA) LI
  STRUCTURE BAPI2
  VALUE(STOCK_DATA) LIKE
  STRUCTURE BAPI2
  VALUE(ACCOUNTING_DATA)
  STRUCTURE BAPI2
  VALUE(USAGE_DECISION_D)
  STRUCTURE BAPI2
  VALUE(CUSTOMER_INCLUDE
  STRUCTURE BAPI2
  VALUE(RETURN) LIKE BAPIR
TABLES
  SYSTEM_STATUS STRUCTU
  USER_STATUS STRUCTU
  MATERIAL_DOCUMENTS ST
*Local data
DATA:
  L_MAPL LIKE MAPL,
  L_PLKO LIKE PLKO,
  L_QALS LIKE QALS,
  L_QAVE LIKE QAVE,
  L_QAMB LIKE QAMB,
  L_QAMB_TAB LIKE QAMB OCC
  L_MSGV1 LIKE SY-MSGV1,
  L_SUBRC LIKE SY-SUBRC.
* Initialize tables passed via interface
CLEAR: SYSTEM_STATUS, USER_ST
REFRESH: SYSTEM_STATUS, USER_
* Check import parameters and authori
*Inspection lot number passed?
IF NUMBER IS INITIAL.
  CALL FUNCTION 'BALW_BAPIRETU
  EXPORTING
    TYPE = C_ERROR
    CL = C_MSGID_QG
    NUMBER = 201
  IMPORTING
    BAPIRETURN = RETURN
  EXCEPTIONS
    OTHERS = 0.
EXIT.
ENDIF.
*Check authority
*/-QA03
L_MSGV1 = C_TCODE_QA03.
CALL FUNCTION 'AUTHORITY_CHECK
  EXPORTING
    TCODE = C_TCODE_QA03
  EXCEPTIONS
    OK = 0
    NOT_OK = 1.
IF NOT SY-SUBRC IS INITIAL.
  CALL FUNCTION 'BALW_BAPIRETU
  EXPORTING
    TYPE = C_ERROR
    CL = C_MSGID_Q5
    NUMBER = 2
    PAR1 = L_MSGV1
  IMPORTING
    BAPIRETURN = RETURN
  EXCEPTIONS
    OTHERS = 0.
EXIT.
ENDIF.
IF USAGE_DECISION_DATA IS REQU
*/-QA13
L_MSGV1 = C_TCODE_QA13.
CALL FUNCTION 'AUTHORITY_CHECK
  EXPORTING
    TCODE = C_TCODE_QA13
  EXCEPTIONS
    OK = 0
    NOT_OK = 1.
IF NOT SY-SUBRC IS INITIAL.
  CALL FUNCTION 'BALW_BAPIRETU
  EXPORTING
    TYPE = C_ERROR
    CL = C_MSGID_Q5
    NUMBER = 2
    PAR1 = L_MSGV1
  IMPORTING
    BAPIRETURN = RETURN
  EXCEPTIONS
    OTHERS = 0.
EXIT.
ENDIF.
* Read requested data
*Read inspection lot
SELECT SINGLE * FROM QALS INTO
WHERE PRUEFLOS = NUMBER.
IF SY-SUBRC <= 0
  *Error: Passed inspection lot number
  L_MSGV1 = NUMBER.
  CALL FUNCTION 'BALW_BAPIRETU
  EXPORTING
    TYPE = C_ERROR
    CL = C_MSGID_QA
    NUMBER = 102
    PAR1 = L_MSGV1
  IMPORTING
    BAPIRETURN = RETURN
  EXCEPTIONS
    OTHERS = 0.
EXIT.
ENDIF.
IF USAGE_DECISION_DATA IS REQU
  *Read usage decision
  SELECT SINGLE * FROM QAVE INT
WHERE PRUEFLOS = NUMBER
AND KART1 = 'S' AND USR_ST =
AND KART2 = 'S' SUB COUNT

```

# Class Converter

## Optimierung des Klassifizierungssystem im SAP S/4HANA und SAP Cloud ERP, private Edition

### Vorteil der Klassifizierung

Das Klassensystem ermöglicht es zusätzliche Daten an Objekten abzulegen. Eingesetzt wird die Klassifizierung bevorzugt bei Materialien, Chargen oder in der Variantenkonfiguration am Material oder im Kundenauftrag. Durch die Bereitstellung von dynamischen Pflegedialogen und die Integration in der Anwendung sind somit Erweiterungen schnell durchgeführt.

Nach Fertigstellung wird das Programm an die Fachabteilung übergeben. Erste Tests zeigen ein neues Problem auf:

#### • Performance

Die Zugriffe auf die Klassifizierung erfolgen sequentiell. Die Bereitstellung der Daten erfordert das Lesen vieler Datensätze. Bei einer sehr umfangreichen Nutzung der Klassifizierung sind Auswertungen fast nicht mehr möglich.

### Nachteil der Klassifizierung

Als nächstes benötigt die Fachabteilung eine Auswertung über die Klassifizierungsdaten. Es stellt schnell heraus, dass die Standardauswertungen zur Bewertung von Klassifizierungen nicht ausreichen. Also wird ein eigener Auswertereport erstellt. Die erste Aufwandsschätzung liegt um ein dreifaches höher als normalerweise üblich. Warum?

- Der einfache Zugriff über ein **Select**-Befehl ist nicht möglich
- Datenelemente und Domänen sind nicht vorhanden und müssen angelegt werden
- Der Zugriff erfolgt umständlich über Funktionen der **Objektsuche**
- Umfangreiche Selektionen wie beispielsweise es die **Select Options** es bieten sind nicht möglich
- Die Ergebnisse der Klassifizierung zu einem Objekt werden als **Tabelle** übergeben und müssen oft erst in eine horizontale Struktur überführt werden.
- Sobald mehrere Sprachen unterstützt werden, müssen die ermittelten Werte über die **Benutzer Einstellungen** konvertiert werden (z.B.: Punkt und Komma Regelung für Dezimalzahlen)
- Das **Know How** der Entwickler für Klassifizierungsdaten muss erst aufgebaut werden

### Optimierung der Klassifizierung

Der Class Converter hebt die Nachteile der Klassifizierung auf. Die Klassifizierungsdaten werden zusätzlich in kundeneigenen Tabellen fortgeschrieben. Mit der Verbuchung eines klassifizierten Objektes wird automatisch genau ein Datensatz erzeugt.

Die Vorteile liegen auf der Hand. Jeder Anwendungsentwickler kann direkt auf die Tabellen zugreifen. ABAP/4 Programme werden kürzer und übersichtlicher. Die Realisierungsaufwände sinken. Durch Setzen von Indices auf diese Tabellen erfolgt der Zugriff in gewohntem Zeitverhalten.

### Funktionsweise

Als erstes werden die Kundeneigenen Tabellen angelegt. Über Customizing Tabellen wird gesteuert, welche Klassifizierungsarten berücksichtigt und in welche Tabellen die Daten gespeichert werden sollen.

Beim Speichern der Klassifizierung werden die Merkmalsname analysiert und mit den Feldnamen der Tabelle abgeglichen. Bei Übereinstimmung wird der Wert des Merkmals in das Tabellenfeld geschrieben. Ist ein Datensatz bereits vorhanden, wird der entsprechende Datensatz überschrieben.

