Mandatory Steps to Prepare Your ABAP Code for SAP HANA

Michael Goedecke and Students who Study at SAP

January 2016

PUBLIC

Latest Version of this presentation can be found in SCN:

http://scn.sap.com/community/abap/hana/blog/2016/01/18/mandatory-steps-to-adapt-abap-code-for-sap-hana

SAP HANA Partner Engineers

This presentations is sponsored by:

SAP HANA Partner Engineers in the Department of
SAP Partner Innovation Lifecycle Services
click here to watch our **YouTube** video

AND

SAP Vocational Education Department
Studying at SAP / Ausbildung bei SAP

Click here to get more infos

WE NEED YOUR FEEBACK

- it will help the students that worked on this presentation to get good grades and a permanent job at SAP
- sponsor more presentation like this on other topics

If this presentation was helpful to you, please sent an email with your feedback to Michael.Goedecke@SAP.com

Introduction

In general existing ABAP code runs on SAP HANA as before.

Only if ABAP code relies on technical specifics on a old database, ABAP code changes might be necessary. This presentation contains the fundamental steps to successfully prepare your code for SAP HANA (see SAP Note: 1785057).

All changes that may be needed are valid for HANA and any old DB.

Therefore it is not needed to distinguish between code for old databases and HANA. The new coding lines will support HANA and traditional databases as well.

HANA is SQL-compliant as any other database.

Agenda

Part 1: What needs to be done?

?

Which of my ABAP code must be changed to avoid potential functional issues?

- Migration and Upgrade Overview
- SELECT Statements
- Native SQL, Hints, ADBC
- Pool Tables, Cluster Tables
- Indices
- Rare Issues

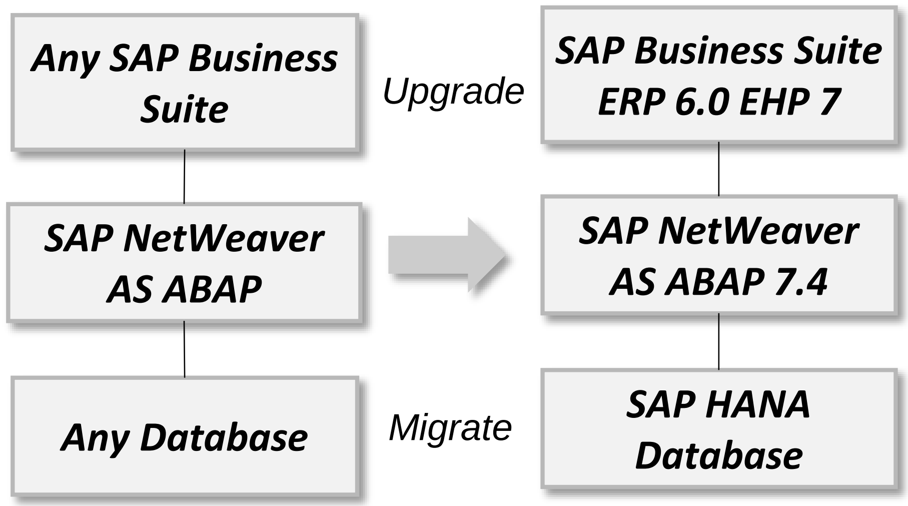
Part 2: How to do it!

!

Use ABAP Code Inspector and Runtime Check Monitor to automatically find potential functional issues.

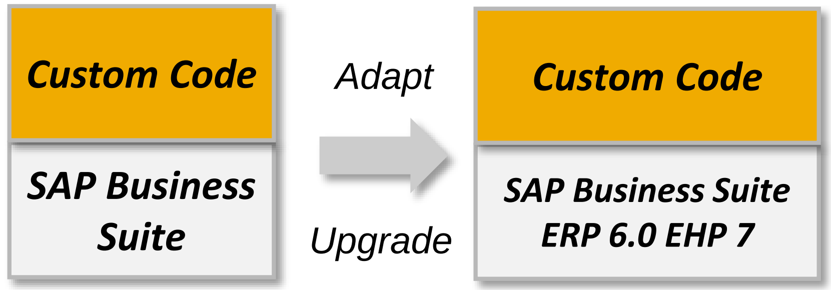
Migrate to SAP Business Suite powered by SAP HANA

SAP Business Suite on SAP HANA



based on **ABAP 7.4** = the “go-to” release for all SAP HANA based ABAP applications

Custom code is a central part of the business functionality



This presentation will focus on the adaption of your custom code

Learn more about AS ABAP 7.4 on http://scn.sap.com/docs/DOC-41400

SELECT Statements in HANA I

Why could changes be required?

→ HANA follows the SQL standard. Implicit behaviors of databases are not guaranteed by the SQL standard

There is a commonly held view that SELECT statements return results implicitly sorted by the used index.
Another commonly held view is that SELECT statements on pool / cluster tables will always be returned sorted by primary key.
This is not guaranteed by SQL standard. The SQL standard requires an ORDER BY statement to return result sets sorted.

Programmers may not be aware that a sorted result set is **not** expectable without an ORDER BY statement. Rather they relied on the usual behavior of previous database systems. However these systems also do not guarantee such behavior. It is simply due to their physical base. HANA will not return result sets sorted without ORDER BY.

SELECT Statements in HANA II

Depending on your code there will be three possibilities what needs to be done.

Possibility 1 & 2

Coding doesn‘t rely on sorted results

Existing SELECT statements are explicitly sorted by an ORDER BY command

Nothing needs to be done

Possibility 3

Code relies on sorted result sets but SELECT statement does not contain an ORDER BY command

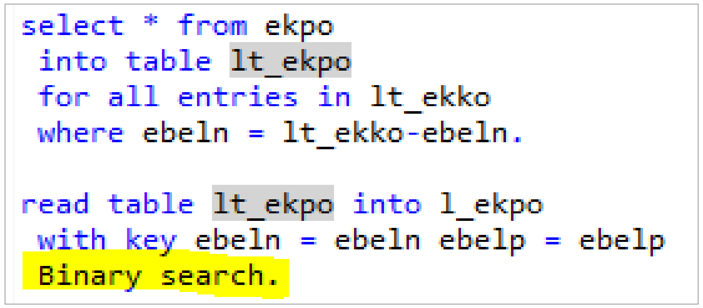
TO-DO

Add an ABAP SORT, an ORDER BY statement or change the code that depends on the sorting.

→ ABAP Code Inspector will help you to find relevant select statements.

SELECT Statements in HANA - Example

This ABAP code relies on sorted DB query results:

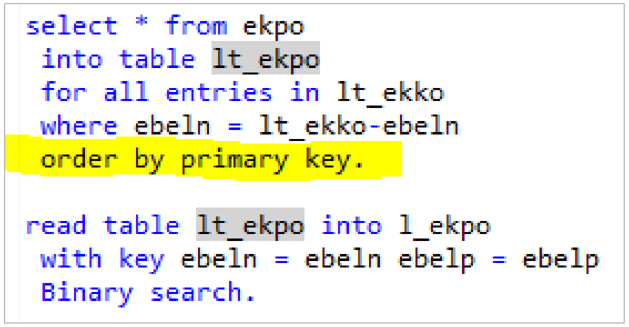


```
select * from ekpo
 into table lt_ekpo
 for all entries in lt_ekko
 where ebeln = lt_ekko-ebeln.

read table lt_ekpo into l_ekpo
 with key ebeln = ebeln ebelp = ebelp
 Binary search.
```

- A “BINARY SEARCH” statement works only for a correctly sorted internal table.
- In this case there is no explicit sorting of the query result done.

→ Error in code which may work by chance because of DB Index usage



```
select * from ekpo
 into table lt_ekpo
 for all entries in lt_ekko
 where ebeln = lt_ekko-ebeln
 order by primary key.

read table lt_ekpo into l_ekpo
 with key ebeln = ebeln ebelp = ebelp
 Binary search.
```

TO-DO

Add explicit sorting by adding ORDER BY.

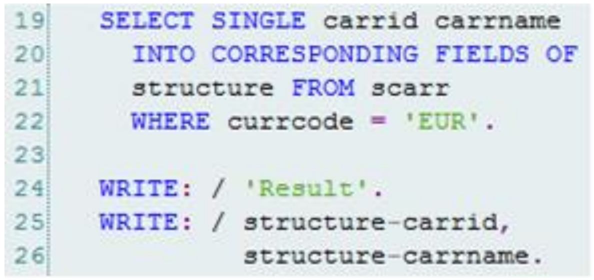
Depending on the situation it may make sense to remove the binary search instead.

SELECT SINGLE Statements in HANA - Example

If you use a SELECT SINGLE statement with a non unique WHERE condition, the result is not guranteed to be unambiguous!

Example:

This is an example of a SINGLE SELECT statement where the result is not unambiguous.



```
SELECT SINGLE carrid carrname
  INTO CORRESPONDING FIELDS OF
  structure FROM scarr
  WHERE currcode = 'EUR'.

WRITE: / 'Result'.
WRITE: / structure-carrid,
         structure-carrname.
```

The result could be any of the 5 airlines who uses the “EUR” currency. Even if the result often is the first entry in the table
(In this case: Air Berlin) it is not guaranteed by the SQL standard.

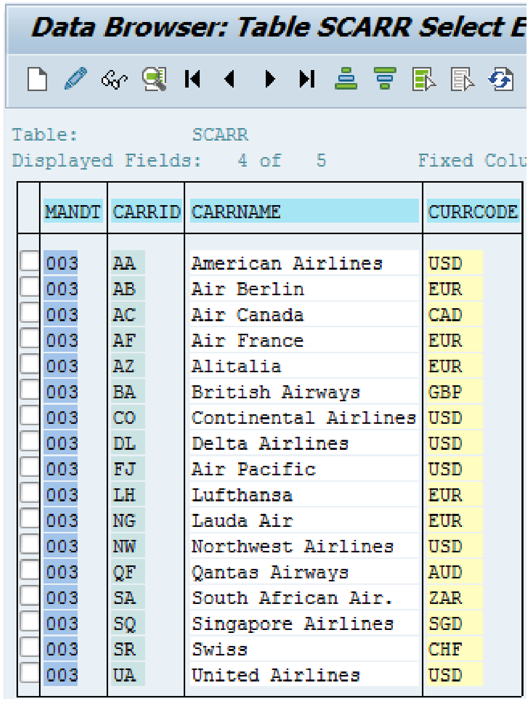


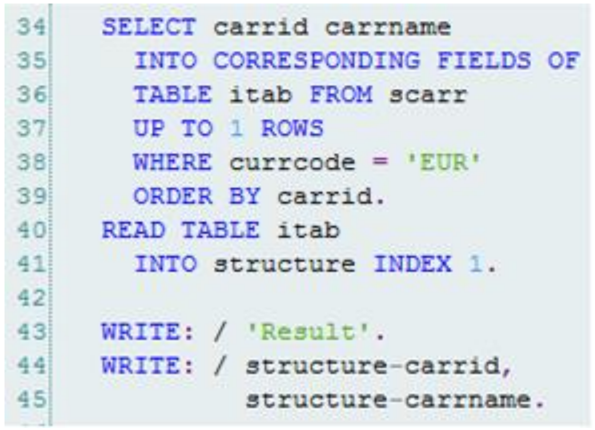
Table: SCARR
Displayed Fields: 4 of 5 Fixed Colu

MANDT	CARRID	CARRNAME	CURRCODE
003	AA	American Airlines	USD
003	AB	Air Berlin	EUR
003	AC	Air Canada	CAD
003	AF	Air France	EUR
003	AZ	Alitalia	EUR
003	BA	British Airways	GBP
003	CO	Continental Airlines	USD
003	DL	Delta Airlines	USD
003	FJ	Air Pacific	USD
003	LH	Lufthansa	EUR
003	NG	Lauda Air	EUR
003	NW	Northwest Airlines	USD
003	QF	Qantas Airways	AUD
003	SA	South African Air.	ZAR
003	SQ	Singapore Airlines	SGD
003	SR	Swiss	CHF
003	UA	United Airlines	USD

The table “scar” contains 5 entries with CURRCODE “EUR”.

SELECT SINGLE Statements in HANA II - Example

Example:
A better way to achieve an unambiguous result could be:



```
SELECT carrid carrname
  INTO CORRESPONDING FIELDS OF
  TABLE itab FROM scarr
  UP TO 1 ROWS
  WHERE currcode = 'EUR'
  ORDER BY carrid.
READ TABLE itab
  INTO structure INDEX 1.

WRITE: / 'Result'.
WRITE: / structure-carrid,
         structure-carrname.
```

Replace the SELECT SINGLE with a SELECT statement.
Add “UP TO 1 ROWS”.
Add “ORDER BY carrid” to guarantee a unambiguous result.
Use the “READ TABLE” to copy your row into a structure.

This workaround is needed because you can’t use
an ORDER BY with an SELECT SINGLE statement.

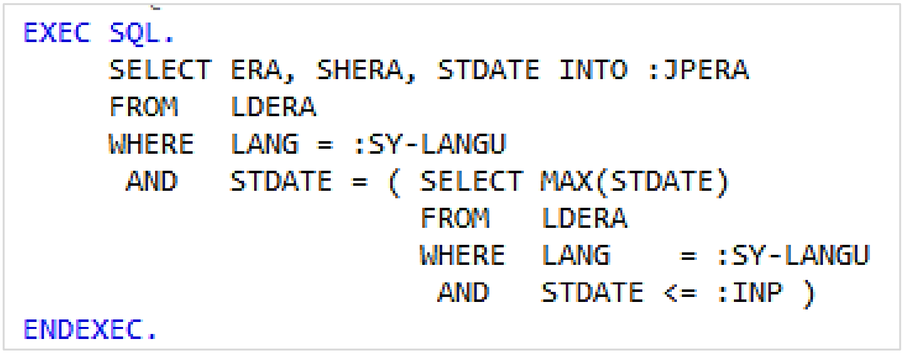
Native SQL, Hints, ADBC

- Native SQL statements embedded in OpenSQL or called via the ADBC interface must be migrated to HANA Native SQL or OpenSQL.
  - Note: With Netweaver 7.40 SP5 OpenSQL got extended, to help to convert native SQL to OpenSQL
- DB hints are DB specific and should be reviewed.

TO-DO

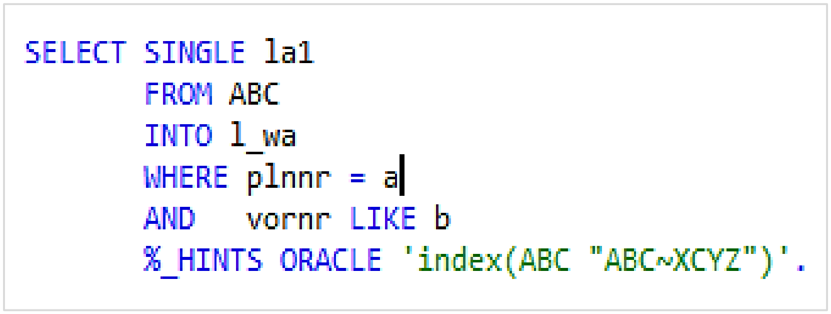
Use OpenSQL if possible, adapt native SQL and DB hints if necessary.

Example for native SQL



```
EXEC SQL.
     SELECT ERA, SHERA, STDATE INTO :JPERA
     FROM   LDERA
     WHERE  LANG = :SY-LANGU
      AND   STDATE = ( SELECT MAX(STDATE)
                       FROM   LDERA
                       WHERE  LANG     = :SY-LANGU
                        AND   STDATE <= :INP )
ENDEXEC.
```

Example for a DB hint



```
SELECT SINGLE la1
       FROM ABC
       INTO l_wa
       WHERE plnnr = a
       AND   vornr LIKE b
       %_HINTS ORACLE 'index(ABC "ABC~XCYZ")'.
```

Pool / Cluster (PC) Tables

- A table pool is a physical table in a DB which can store multiple (logical) pool tables.
- A pool table is a logical table in a table pool.
- A table cluster is a physical table in a DB which can store multiple (logical) cluster tables.
- A cluster table is a logical table in a table cluster

These constructs are obsolete and should not be used anymore.

Learn more about declustering / depooling in OSS note: 1785057

Pool / Cluster (PC) Tables

- Most pool tables and cluster tables of SAP NetWeaver and SAP Business Suite were migrated to transparent tables for HANA.
- In a new ERP installation or a properly migrated ERP system there will be around 30 pool tables left. These pool tables can’t be converted for technical reasons. There won’t be any cluster tables left.
- ABAP code which directly accesses a table pool / table cluster leads to syntax errors.

TO-DO

- Convert your own PC tables to transparent tables.
- Get rid of direct access to table pools / table clusters.
- Don’t create PC tables in the future.

Learn more about declustering / depooling in OSS note: 1785057

Indices in HANA

In most cases, SAP HANA does not require secondary indices for good search performance.

- In SAP HANA data is stored in column store by default. Basically every column is an index.
- Row store should only be used in rare cases (SAP recommendation)
- To reduce main memory consumption, and to improve insert performance all existing non-unique secondary database indices on columnar tables are removed during migration or do not get created during installation.
- Unique indices stay as they represent a constraint on the table.

This is valid for all AS ABAP systems from SAP NetWeaver 7.4 onwards!

Optional Task
Performance Optimization on HANA

As performance improvements are optional, we will give only a brief overview.

Replace ABAP with SQL

- Work directly on the in-memory-data which is the most efficient execution
- Avoid unnecessary round trips to the application server
- Execute as much as possible in parallel by HANA

Create Secondary Indices

- Use them for highly selective queries on non-primary key fields. These queries can be significantly improved by indices on single fields which are most selective.
- You may use SAP Note 1794297 to find and create these indices.

Optional Task
Performance Optimization on HANA

Reduce SELECT statements to the columns you really need.

- Some tables contain a lot of columns (e.g. 100-400)
- In many cases the complete data record is selected although only a few columns are needed
- Reducing the selected columns to a minimum will save time and reduce CPU load

Convert from column store to row store

- Using row store should be the exception because statistical analyses/ aggregates perform better on column store
- Use row store for tables where data is temporarily stored
- Business data should usually go on column store

Rare Issues

DB index analysis

- Analysis of technical DB index information using function modules 'DB_EXISTS_INDEX‘, 'DD_INDEX_NAME’
- Recommendation: Get rid of DB index analysis
- Reason: Most indices are no longer needed and therefore not created in HANA

SQL Compiler error

- New OpenSQL compiler is active as of NW 7.4 SP2
- Compiler identifies syntactically wrong coding which was ignored before
- Ignoring these errors in existing coding leads to build errors or runtime errors
- Risk of having errors is low: Only 10 corrections for the whole ERP were made

SQL Compiler error: For details see SAP note 1832139 - OpenSQL runtime environment

Agenda

Part 1: What needs to be done?

?

Which of my ABAP code must be changed to avoid potential functional issues?

- Migration overview
- SELECT statements
- Native SQL, hints, ADBC
- Pool tables, cluster tables
- Rare issues

Part 2: How to do it

!

Use ABAP Code Inspector and Runtime Check Monitor to automatically find potential functional issues

The SAP Code Inspector – (Transaction SCI)

The tool “SAP Code Inspector” (SCI) can scan and analyze ABAP code.
This tool will help us in this presentation find automatically coding that may need to be changed.

It performs different checks on your ABAP programs/ function groups/ classes etc.

You can choose which checks should be applied.

The Code Inspector only can check Objects which were developed on the system where you use the Code Inspector.
Check the property “Original System” in the Object Dictionary Entry.
Other Objects will be ignored by the Code Inspector, without a warning!

For all main functional error categories we provide Code Inspector checks to automatically find the described issues.

Code Inspector – Short Discription

“Critical Statements” check

- *Find critical Statements such as Native SQL Statements or Database Hints*

“Use of ADBC Interface” check

- Find uses of ADBC classes

“Extended Program Check (SLIN)” check (ONLY NEEDED if SAP NOTE 2251947 is NOT implemented)

- *Find Warnings to find SELECT SINGLE Statements where the result is not unambiguous because the WHERE clause points on a non unique field of a table*

“Search DB Operations in Pool/Cluster Tables” check

- The check finds all SQL accesses to physical table pools or table clusters (Table Pool ≠ Pool Table).
- (There is a mistake in the title. It should be “Table Pools/Clusters”).
- All accesses with SQL to these tables are senseless after depooling/declustering and shall be corrected.

Code Inspector – Short Discription II

“Depooling/Declustering: Search for…w/o ORDER BY” check

- Searches for SELECT <pool/cluster table> without ORDER BY but no further analysis of the data flow is done
- works only for pool/cluster DB tables (before and after depooling/declustering)

“Search problematic statement…w/o ORDER BY” check

- *Preferred check to find code which relies on sorted DB content*
- searches for statements like READ BINARY SEARCH, DELETE ADJACENT DUPLICATES, … accessing unsorted DB content
- works for transparent *and* pool/cluster DB tables

“Find ABAP Statement Patterns” check

- Searches multiple statement patterns.
- Example:
  READ TABLE * WITH KEY +COMP+ = +
  This pattern searches for an ABAP statement that contains any sequence of tokens after READ TABLE, followed by WITH KEY. After that, it expects a token that is precisely six characters long, begins and ends with any character, and has the character sequence COMP at its centre. At the end of the statement, it expects '=' and any word.

Code Inspector – Availability of the checks I

The Code Inspector can perform multiple checks on your ABAP code.
Based on your SAP_BASIS release different checkboxes will be available (see Table below).

Checkbox/ SAP BASIS Release	7.01	7.02	7.31		7.40			7.50
	SP17	≥ SP14	≤ SP04	≥ SP09	≤ SP2	SP3 - SP8	≥ SP9	
Security Checks								
Critical Statements	X	X	X	X	X	X	X	X
Use of ADBC Interface		X	X	X	X	X	X	X
Syntax Check/Generation								
Extended Program Check (SLIN) **(ONLY needed if SAP NOTE 2251947 is NOT implemented)**	X	X	X	X	X	X	X	X
Robust Programming								
Search DB Operations in Pool/Cluster Tables		X		X		X	X	X
Depooling / Declustering: Search SELECT for Pool/Cluster-Tables w/o ORDER BY		X		X	X	X	X[1]	X[1]
Search problematic statements for result of SELECT / OPEN CURSOR without ORDER BY		X		X		X	X	X
Search Functs.								
Search ABAP Statement Patterns		X	X	X		X	X	X

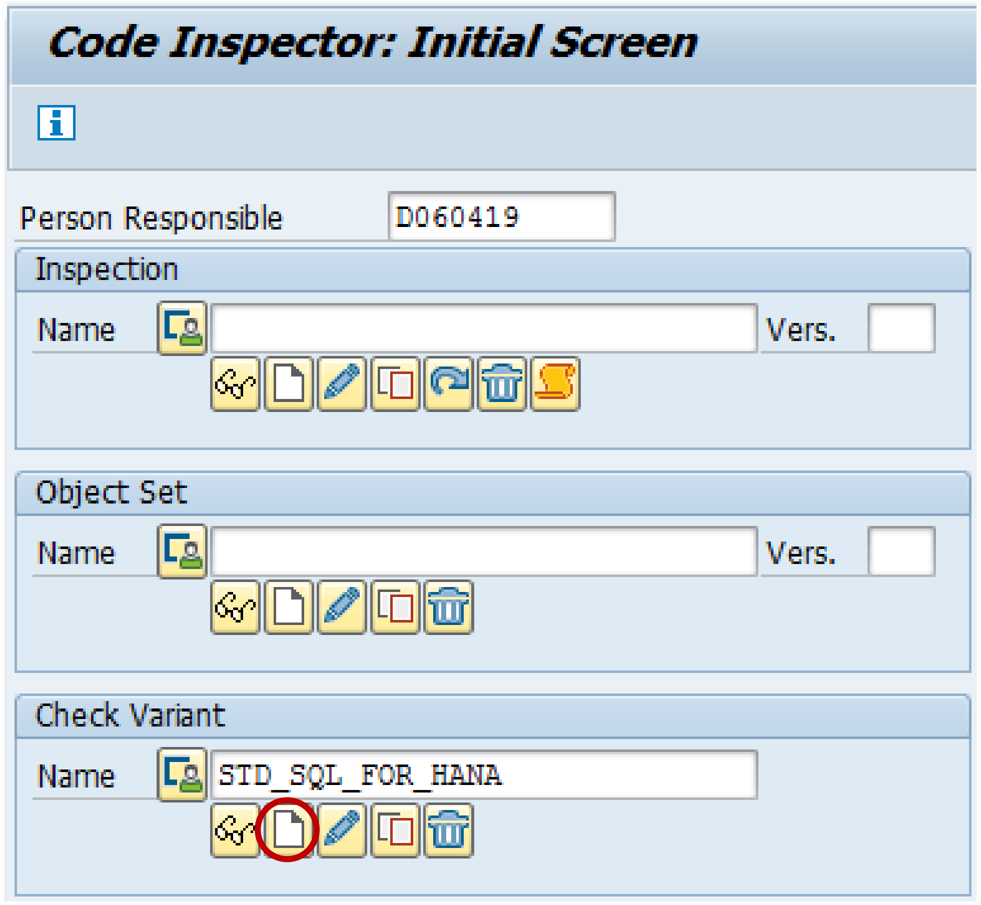
[1]This Checkbox can no longer be found in section “Robust Programming“ but in section “Intern. Tests“

Code Inspector – Availability of the checks II

Please make sure all checks are available for your system. If not, you may need transport nodes in order to add the missing checks.

For example: If you have SAP_BASIS 740 SP02 you might need the transport SAP Note 1875529 to add the “Search problematic statements for result of SELECT / OPEN CURSOR without ORDER BY” check.

Code Inspector – Step By Step



Code Inspector: Initial Screen

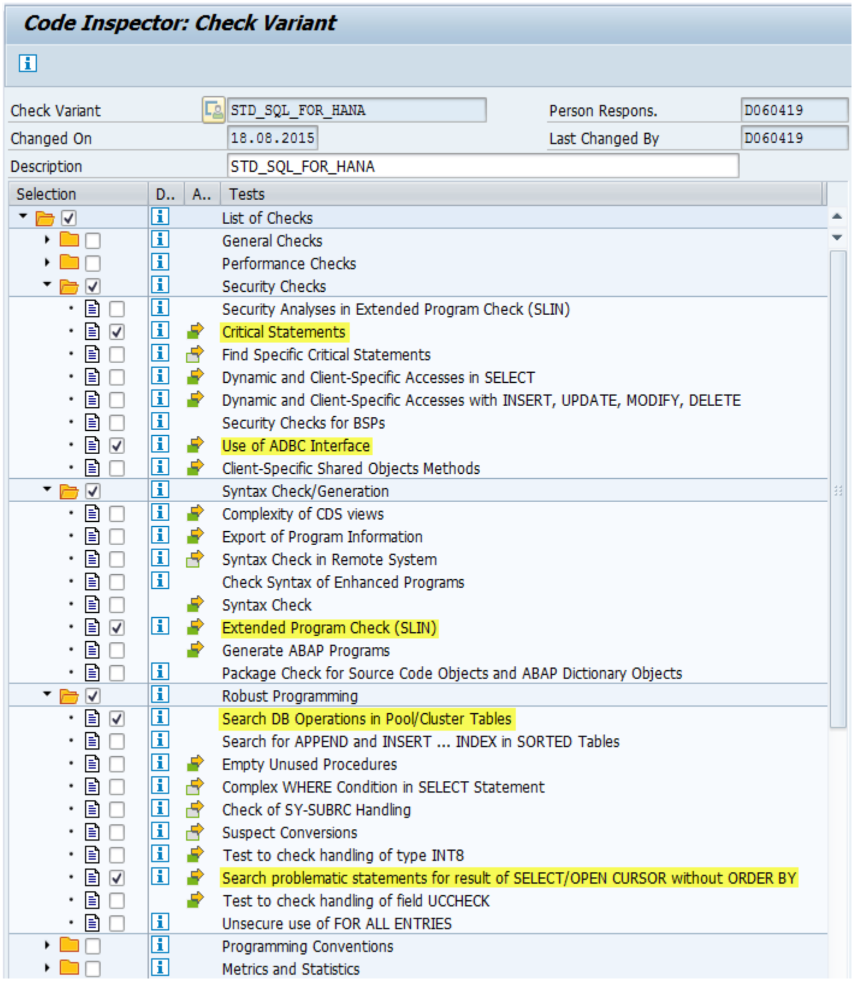
Person Responsible	D060419		
Inspection			
Name		Vers.	
Object Set			
Name		Vers.	
Check Variant			
Name	STD_SQL_FOR_HANA		

Step 1: Use transaction SCI to open the Code Inspector.

Step 2: Enter the text "STD_SQL_FOR_HANA" for a new check variant and press "Create".

Code Inspector – Step By Step

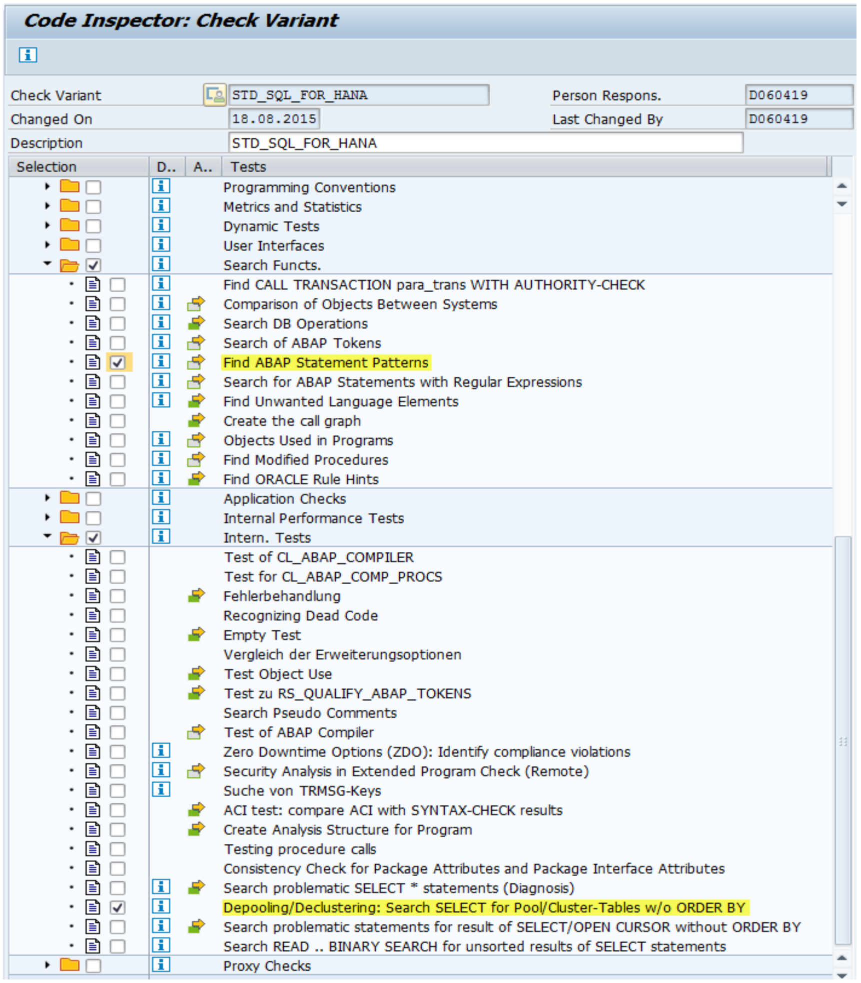
Step 3 for SAP_BASIS 740 SP 9 or 750: Mark the following check boxes as shown in the screenshot.



Code Inspector: Check Variant

Check Variant	STD_SQL_FOR_HANA	Person Respons.	D060419
Changed On	18.08.2015	Last Changed By	D060419
Description	STD_SQL_FOR_HANA		

Selection	D..	A..	Tests
▾ ✓	ℹ		List of Checks
▸ ☐	ℹ		General Checks
▸ ☐	ℹ		Performance Checks
▾ ✓	ℹ		Security Checks
• ☐	ℹ		Security Analyses in Extended Program Check (SLIN)
• ✓	ℹ	➦	Critical Statements
• ☐	ℹ	➦	Find Specific Critical Statements
• ☐	ℹ	➦	Dynamic and Client-Specific Accesses in SELECT
• ☐	ℹ	➦	Dynamic and Client-Specific Accesses with INSERT, UPDATE, MODIFY, DELETE
• ☐	ℹ		Security Checks for BSPs
• ✓	ℹ	➦	Use of ADBC Interface
• ☐	ℹ	➦	Client-Specific Shared Objects Methods
▾ ✓	ℹ		Syntax Check/Generation
• ☐	ℹ	➦	Complexity of CDS views
• ☐	ℹ	➦	Export of Program Information
• ☐	ℹ	➦	Syntax Check in Remote System
• ☐	ℹ		Check Syntax of Enhanced Programs
• ☐		➦	Syntax Check
• ✓	ℹ	➦	Extended Program Check (SLIN)
• ☐		➦	Generate ABAP Programs
• ☐	ℹ		Package Check for Source Code Objects and ABAP Dictionary Objects
▾ ✓	ℹ		Robust Programming
• ✓	ℹ		Search DB Operations in Pool/Cluster Tables
• ☐	ℹ		Search for APPEND and INSERT ... INDEX in SORTED Tables
• ☐	ℹ	➦	Empty Unused Procedures
• ☐	ℹ	➦	Complex WHERE Condition in SELECT Statement
• ☐	ℹ	➦	Check of SY-SUBRC Handling
• ☐	ℹ	➦	Suspect Conversions
• ☐	ℹ	➦	Test to check handling of type INT8
• ✓	ℹ	➦	Search problematic statements for result of SELECT/OPEN CURSOR without ORDER BY
• ☐		➦	Test to check handling of field UCCHECK
• ☐	ℹ		Unsecure use of FOR ALL ENTRIES
▸ ☐	ℹ		Programming Conventions
▸ ☐	ℹ		Metrics and Statistics



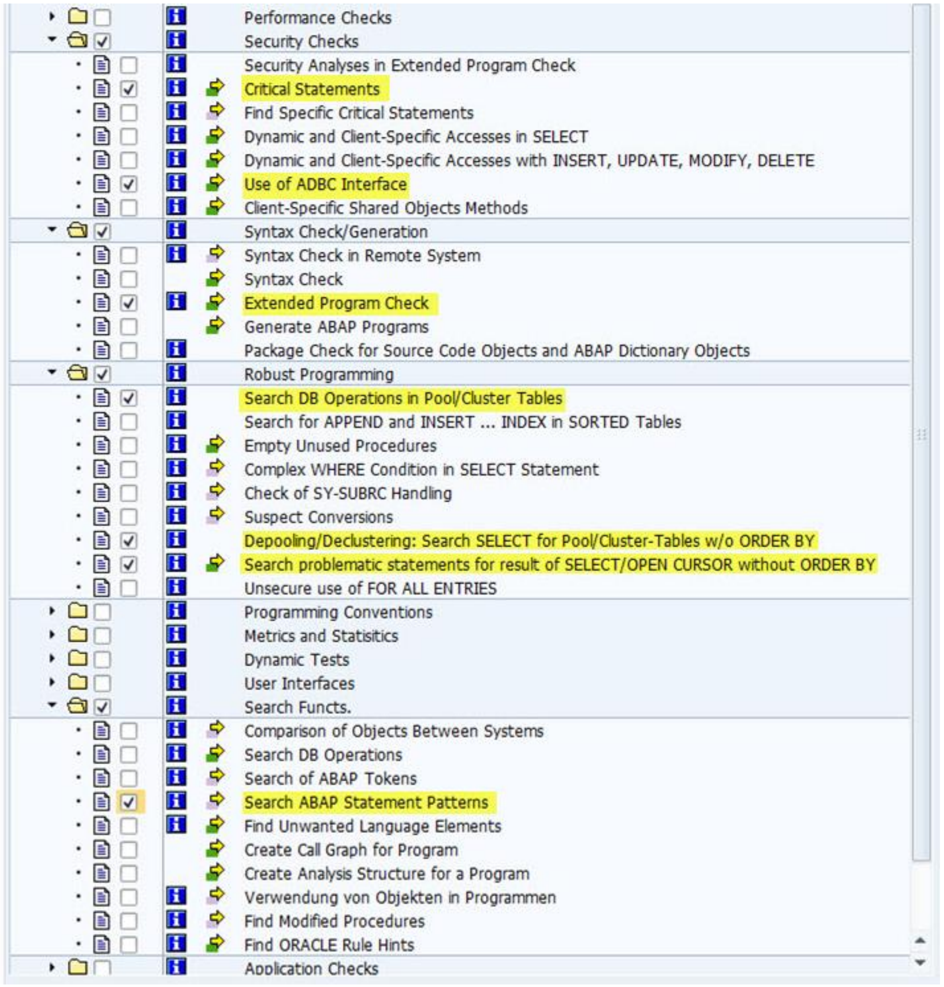
Code Inspector: Check Variant

Check Variant	STD_SQL_FOR_HANA	Person Respons.	D060419
Changed On	18.08.2015	Last Changed By	D060419
Description	STD_SQL_FOR_HANA		

Selection	D..	A..	Tests
▸ ☐	ℹ		Programming Conventions
▸ ☐	ℹ		Metrics and Statistics
▸ ☐	ℹ		Dynamic Tests
▸ ☐	ℹ		User Interfaces
▾ ✓	ℹ		Search Functs.
• ☐	ℹ		Find CALL TRANSACTION para_trans WITH AUTHORITY-CHECK
• ☐	ℹ	➦	Comparison of Objects Between Systems
• ☐	ℹ	➦	Search DB Operations
• ☐	ℹ	➦	Search of ABAP Tokens
• ✓	ℹ	➦	Find ABAP Statement Patterns
• ☐	ℹ	➦	Search for ABAP Statements with Regular Expressions
• ☐	ℹ	➦	Find Unwanted Language Elements
• ☐		➦	Create the call graph
• ☐	ℹ	➦	Objects Used in Programs
• ☐	ℹ	➦	Find Modified Procedures
• ☐	ℹ	➦	Find ORACLE Rule Hints
▸ ☐	ℹ		Application Checks
▸ ☐	ℹ		Internal Performance Tests
▾ ✓	ℹ		Intern. Tests
• ☐			Test of CL_ABAP_COMPILER
• ☐			Test for CL_ABAP_COMP_PROCS
• ☐		➦	Fehlerbehandlung
• ☐			Recognizing Dead Code
• ☐		➦	Empty Test
• ☐			Vergleich der Erweiterungsoptionen
• ☐		➦	Test Object Use
• ☐		➦	Test zu RS_QUALIFY_ABAP_TOKENS
• ☐			Search Pseudo Comments
• ☐		➦	Test of ABAP Compiler
• ☐	ℹ		Zero Downtime Options (ZDO): Identify compliance violations
• ☐	ℹ	➦	Security Analysis in Extended Program Check (Remote)
• ☐	ℹ		Suche von TRMSG-Keys
• ☐		➦	ACI test: compare ACI with SYNTAX-CHECK results
• ☐		➦	Create Analysis Structure for Program
• ☐			Testing procedure calls
• ☐			Consistency Check for Package Attributes and Package Interface Attributes
• ☐	ℹ	➦	Search problematic SELECT * statements (Diagnosis)
• ✓	ℹ		Depooling/Declustering: Search SELECT for Pool/Cluster-Tables w/o ORDER BY
• ☐	ℹ	➦	Search problematic statements for result of SELECT/OPEN CURSOR without ORDER BY
• ☐	ℹ		Search READ .. BINARY SEARCH for unsorted results of SELECT statements
▸ ☐	ℹ		Proxy Checks

Code Inspector – Step By Step

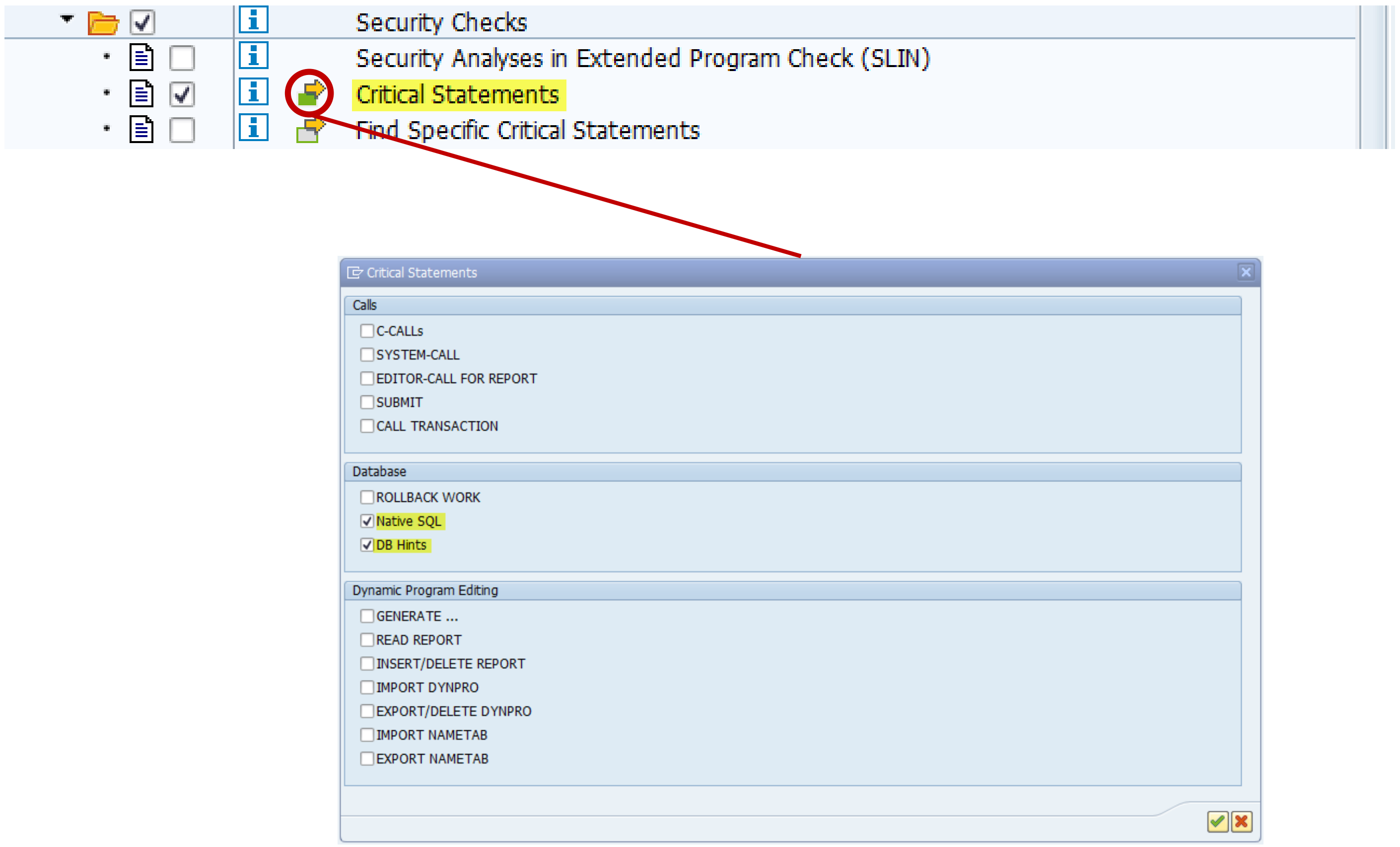
Step 3 for SAP_BASIS 740 SP 8 and lower: Mark the following check boxes as shown in the screenshot.



- [ ] Performance Checks
- [✓] Security Checks
  - [ ] Security Analyses in Extended Program Check
  - [✓] Critical Statements
  - [ ] Find Specific Critical Statements
  - [ ] Dynamic and Client-Specific Accesses in SELECT
  - [ ] Dynamic and Client-Specific Accesses with INSERT, UPDATE, MODIFY, DELETE
  - [✓] Use of ADBC Interface
  - [ ] Client-Specific Shared Objects Methods
- [✓] Syntax Check/Generation
  - [ ] Syntax Check in Remote System
  - [ ] Syntax Check
  - [✓] Extended Program Check
  - [ ] Generate ABAP Programs
  - [ ] Package Check for Source Code Objects and ABAP Dictionary Objects
- [✓] Robust Programming
  - [✓] Search DB Operations in Pool/Cluster Tables
  - [ ] Search for APPEND and INSERT ... INDEX in SORTED Tables
  - [ ] Empty Unused Procedures
  - [ ] Complex WHERE Condition in SELECT Statement
  - [ ] Check of SY-SUBRC Handling
  - [ ] Suspect Conversions
  - [✓] Depooling/Declustering: Search SELECT for Pool/Cluster-Tables w/o ORDER BY
  - [✓] Search problematic statements for result of SELECT/OPEN CURSOR without ORDER BY
  - [ ] Unsecure use of FOR ALL ENTRIES
- [ ] Programming Conventions
- [ ] Metrics and Statisitics
- [ ] Dynamic Tests
- [ ] User Interfaces
- [✓] Search Functs.
  - [ ] Comparison of Objects Between Systems
  - [ ] Search DB Operations
  - [ ] Search of ABAP Tokens
  - [✓] Search ABAP Statement Patterns
  - [ ] Find Unwanted Language Elements
  - [ ] Create Call Graph for Program
  - [ ] Create Analysis Structure for a Program
  - [ ] Verwendung von Objekten in Programmen
  - [ ] Find Modified Procedures
  - [ ] Find ORACLE Rule Hints
- [ ] Application Checks

Code Inspector – Step By Step

Step 4: Click on the Arrow with the green box next to “Critical Statements”. Mark the shown checkboxes within the Pop-Up.



[✓] Security Checks
[ ] Security Analyses in Extended Program Check (SLIN)
[✓] Critical Statements
[ ] Find Specific Critical Statements

Critical Statements

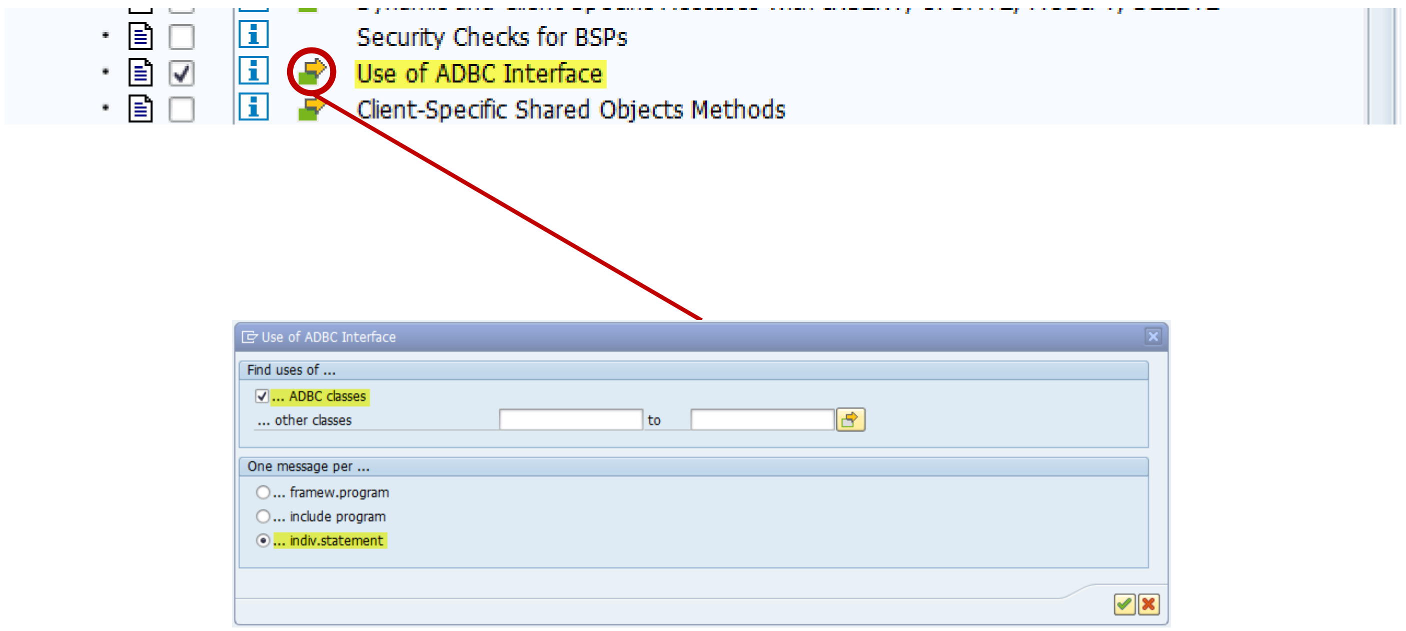
Calls
- [ ] C-CALLs
- [ ] SYSTEM-CALL
- [ ] EDITOR-CALL FOR REPORT
- [ ] SUBMIT
- [ ] CALL TRANSACTION

Database
- [ ] ROLLBACK WORK
- [✓] Native SQL
- [✓] DB Hints

Dynamic Program Editing
- [ ] GENERATE ...
- [ ] READ REPORT
- [ ] INSERT/DELETE REPORT
- [ ] IMPORT DYNPRO
- [ ] EXPORT/DELETE DYNPRO
- [ ] IMPORT NAMETAB
- [ ] EXPORT NAMETAB

Code Inspector – Step By Step

Step 5: Click on the Arrow with the green box next to “Use of ADBC Interface”. Mark the shown checkboxes within the Pop-Up.


- Security Checks for BSPs
- ✓ Use of ADBC Interface
- Client-Specific Shared Objects Methods

Use of ADBC Interface

Find uses of ...

- ✓ ... ADBC classes
- ... other classes to

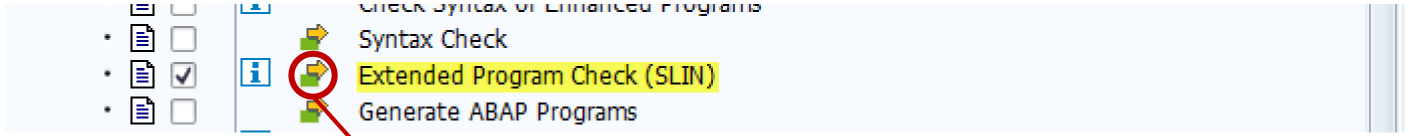
One message per ...

- ○ ... framew.program
- ○ ... include program
- ● ... indiv.statement

Code Inspector – Step By Step

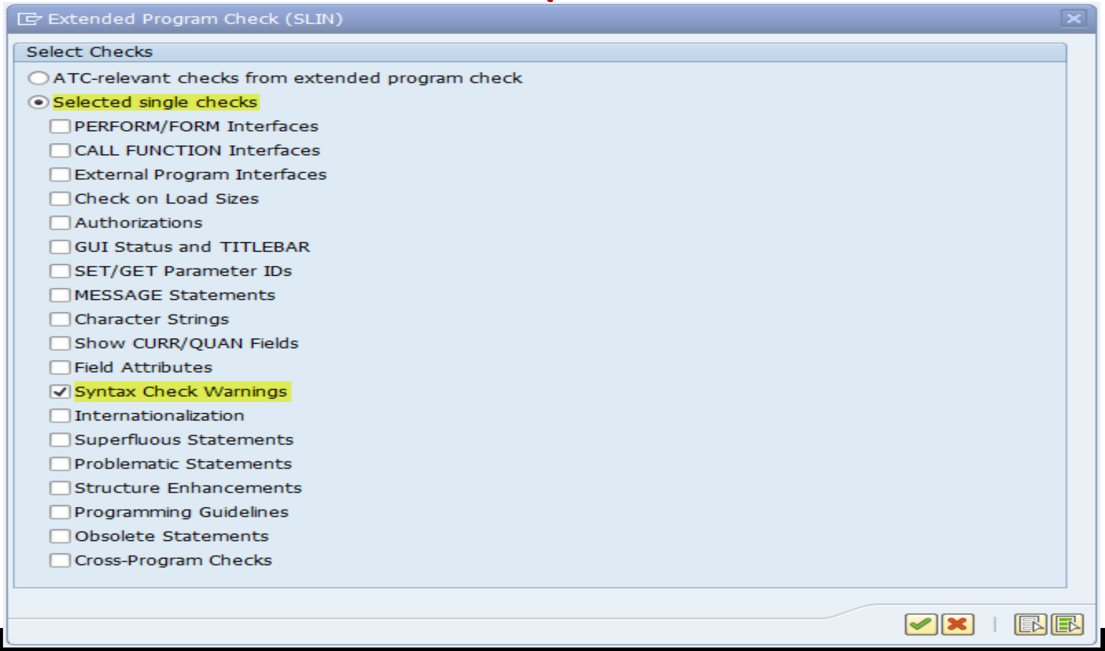
(ONLY NEEDED if SAP NOTE 2251947 is NOT implemented)

Step 6: Click on the Arrow with the green box next to “Extended Program Check (SLIN)”. Mark the shown checkboxes within the Pop-Up.



☐ Syntax Check
☑ Extended Program Check (SLIN)
☐ Generate ABAP Programs

(SAP NOTE 2251947 will add the relevant checks to the „Order By“ checks), which makes then this check superflous.



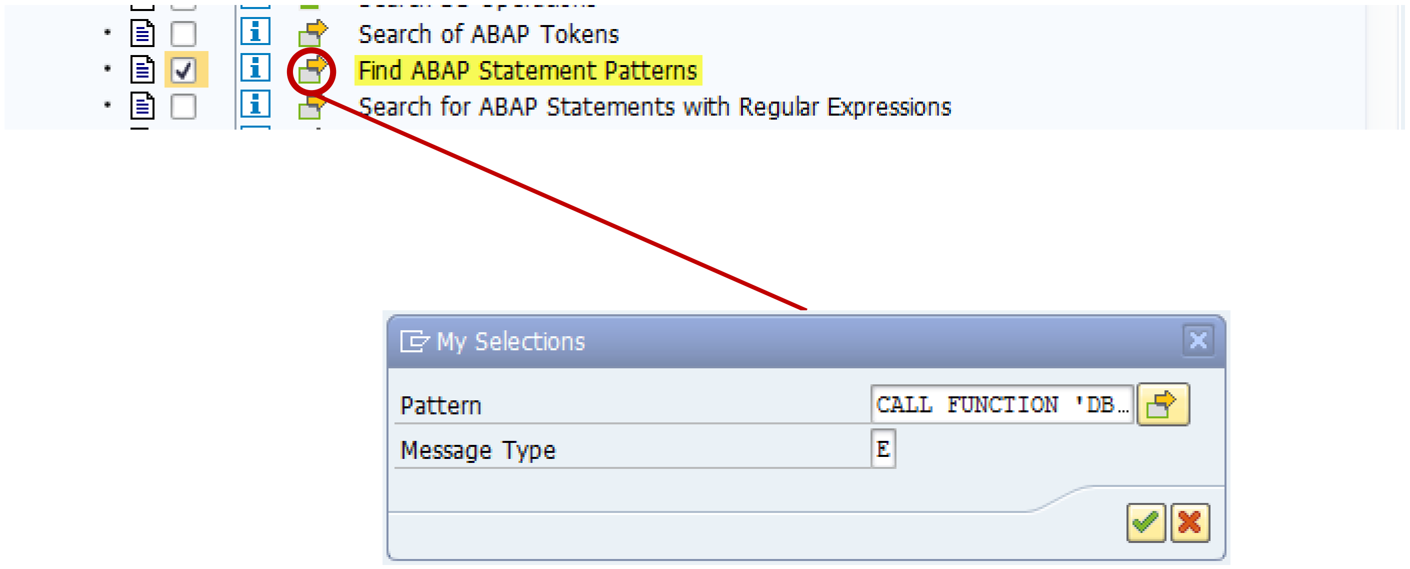
Extended Program Check (SLIN)

Select Checks

○ ATC-relevant checks from extended program check
◉ Selected single checks
☐ PERFORM/FORM Interfaces
☐ CALL FUNCTION Interfaces
☐ External Program Interfaces
☐ Check on Load Sizes
☐ Authorizations
☐ GUI Status and TITLEBAR
☐ SET/GET Parameter IDs
☐ MESSAGE Statements
☐ Character Strings
☐ Show CURR/QUAN Fields
☐ Field Attributes
☑ Syntax Check Warnings
☐ Internationalization
☐ Superfluous Statements
☐ Problematic Statements
☐ Structure Enhancements
☐ Programming Guidelines
☐ Obsolete Statements
☐ Cross-Program Checks

Code Inspector – Step By Step

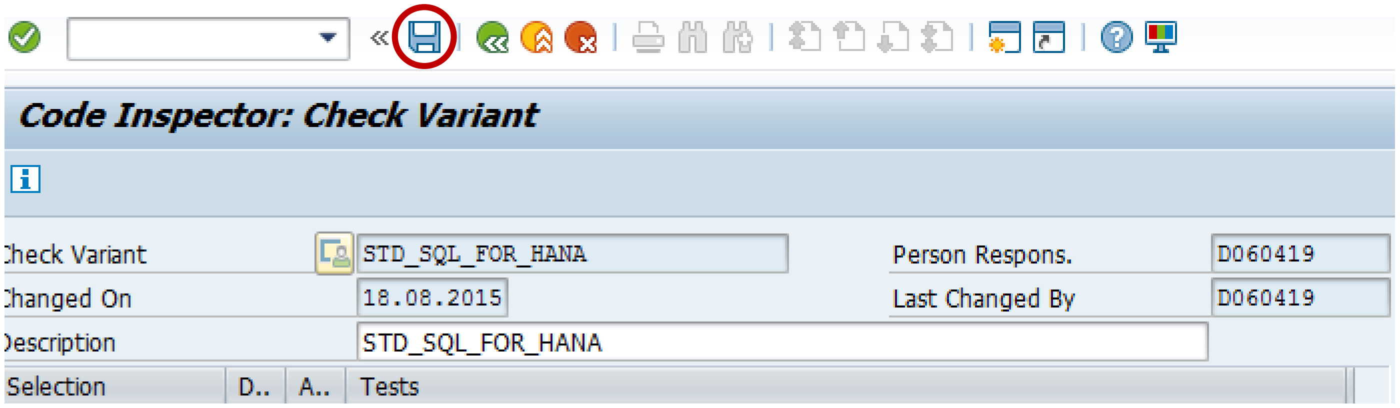
Step 7: Click on the Arrow with the green box next to “Find ABAP Statement Patterns”. Enter “CALL FUNCTION 'DB_EXISTS_INDEX' * “ in the Pattern field and type “E“ for Message Type (Without “ “).


Search of ABAP Tokens
Find ABAP Statement Patterns
Search for ABAP Statements with Regular Expressions
My Selections

Pattern	CALL FUNCTION 'DB…
Message Type	E

Code Inspector – Step By Step

Step 8: If you are finished, press “Save” in order to save your variant.

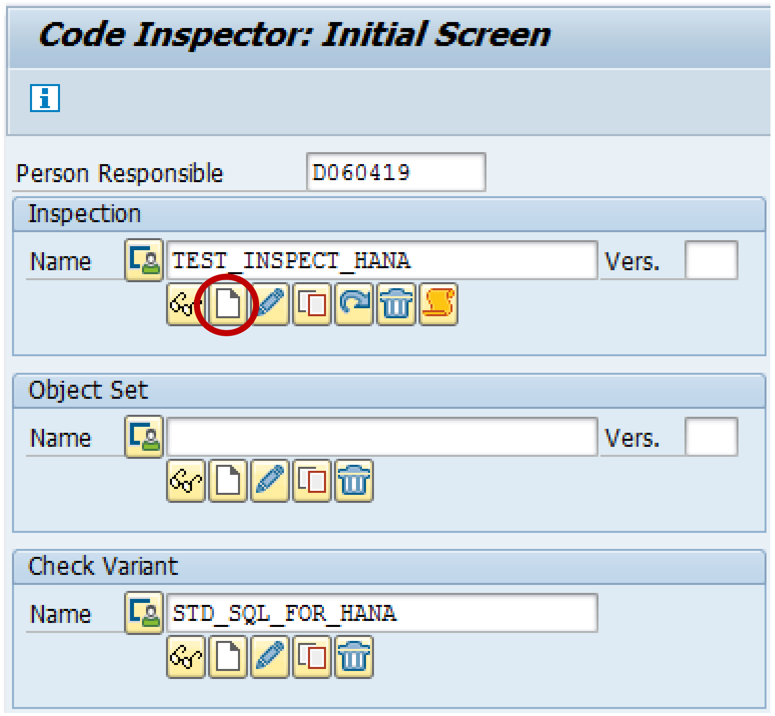


Code Inspector: Check Variant

heck Variant	STD_SQL_FOR_HANA	Person Respons.	D060419
hanged On	18.08.2015	Last Changed By	D060419
escription	STD_SQL_FOR_HANA		

Selection	D..	A..	Tests

Code Inspector – Step By Step



Code Inspector: Initial Screen

Person Responsible: D060419

Inspection

Name: TEST_INSPECT_HANA Vers.:

Object Set

Name: Vers.:

Check Variant

Name: STD_SQL_FOR_HANA

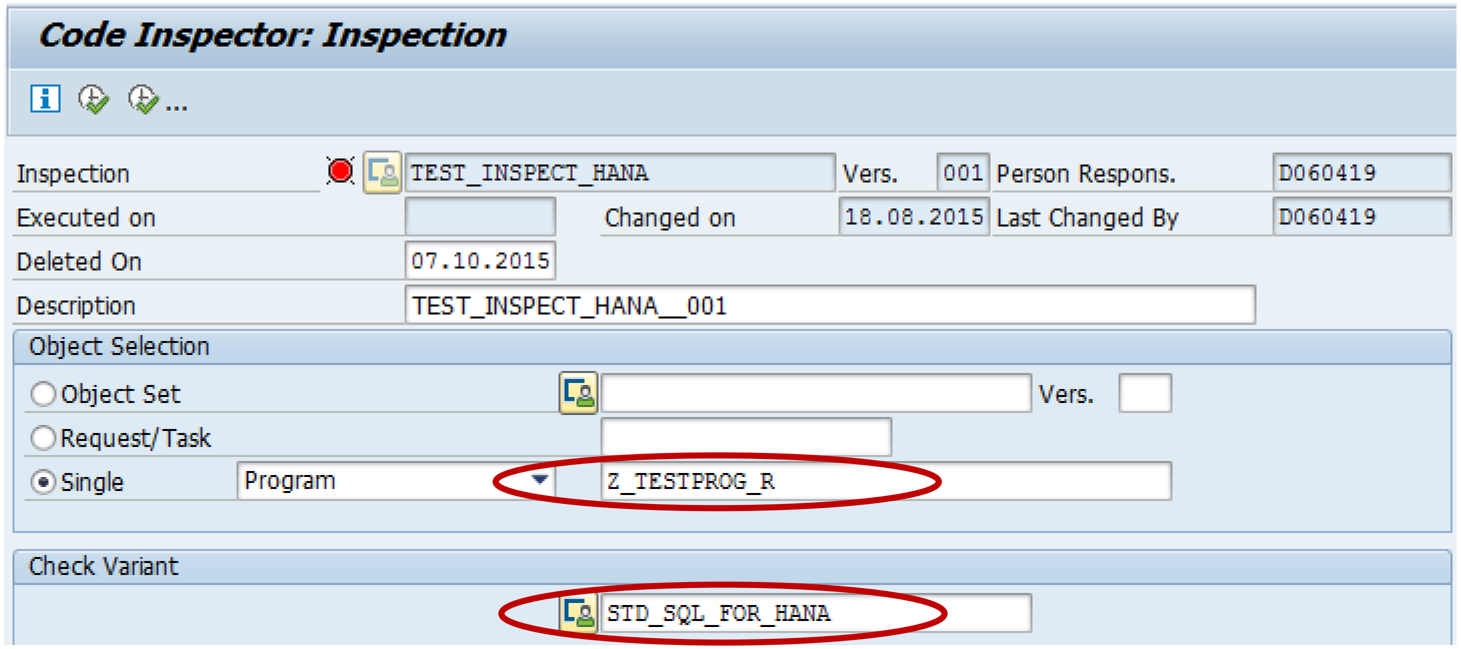
Step 9: Go back to the initial code inspector screen (SCI) and create an inspection.

Code Inspector – Step By Step

Step 10: Add the object(s) you want to check in the section "Object Selection". You can either choose single objects or an object set. Object sets can be created in the initial Code Inspector (SCI) screen.

Step 11: Add the previously created check variant.

Step 12: Execute the code inspection.



Code Inspector: Inspection

Inspection	TEST_INSPECT_HANA	Vers. 001	Person Respons. D060419
Executed on		Changed on 18.08.2015	Last Changed By D060419
Deleted On	07.10.2015		
Description	TEST_INSPECT_HANA__001		

Object Selection

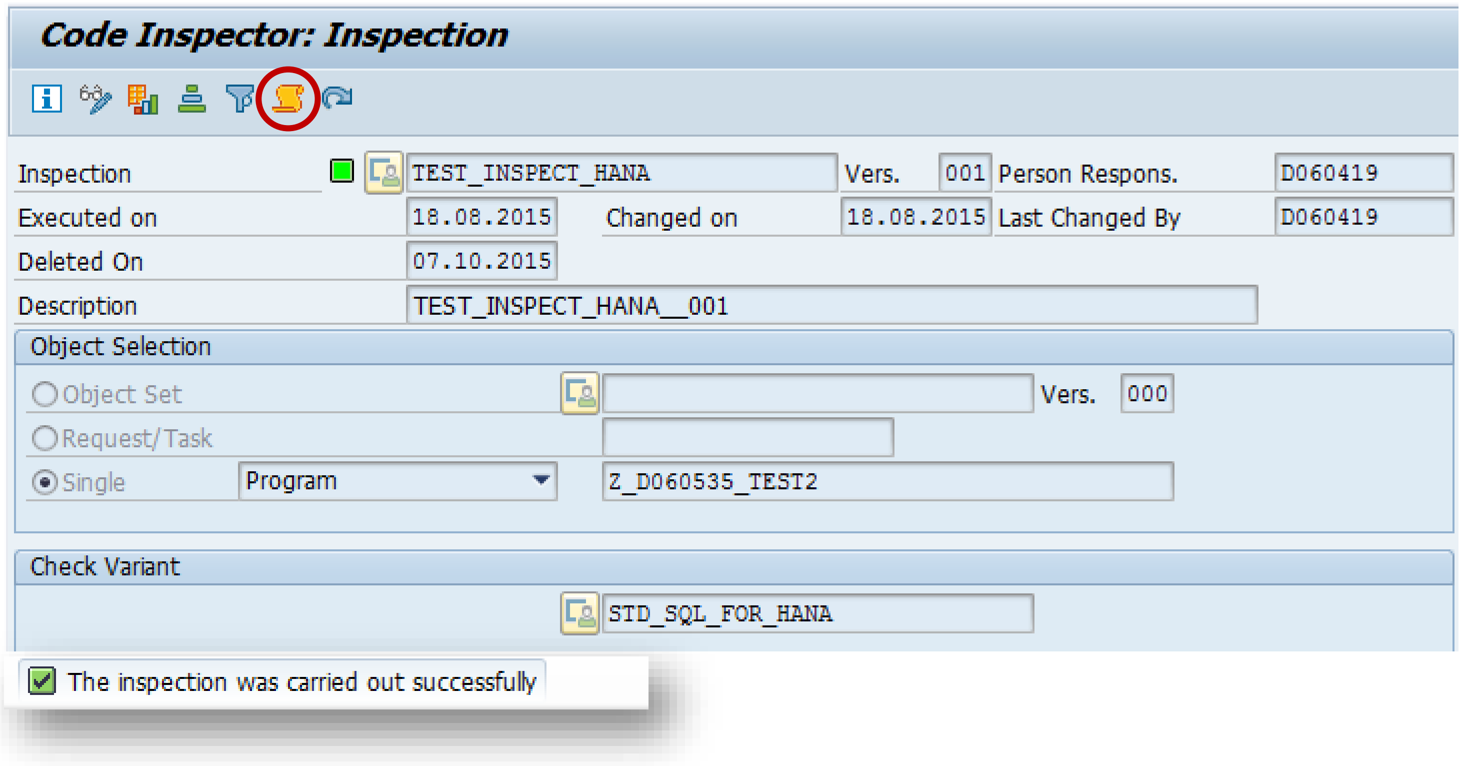
○ Object Set		Vers.
○ Request/Task		
● Single	Program	Z_TESTPROG_R

Check Variant

STD_SQL_FOR_HANA

Code Inspector – Step By Step

Step 13: View the results of the successful code inspection by clicking the marked icon.



Code Inspector: Inspection

Inspection	🟩 TEST_INSPECT_HANA	Vers. 001	Person Respons. D060419
Executed on	18.08.2015	Changed on 18.08.2015	Last Changed By D060419
Deleted On	07.10.2015		
Description	TEST_INSPECT_HANA__001		

Object Selection

- ○ Object Set — Vers. 000
- ○ Request/Task
- ● Single — Program — Z_D060535_TEST2

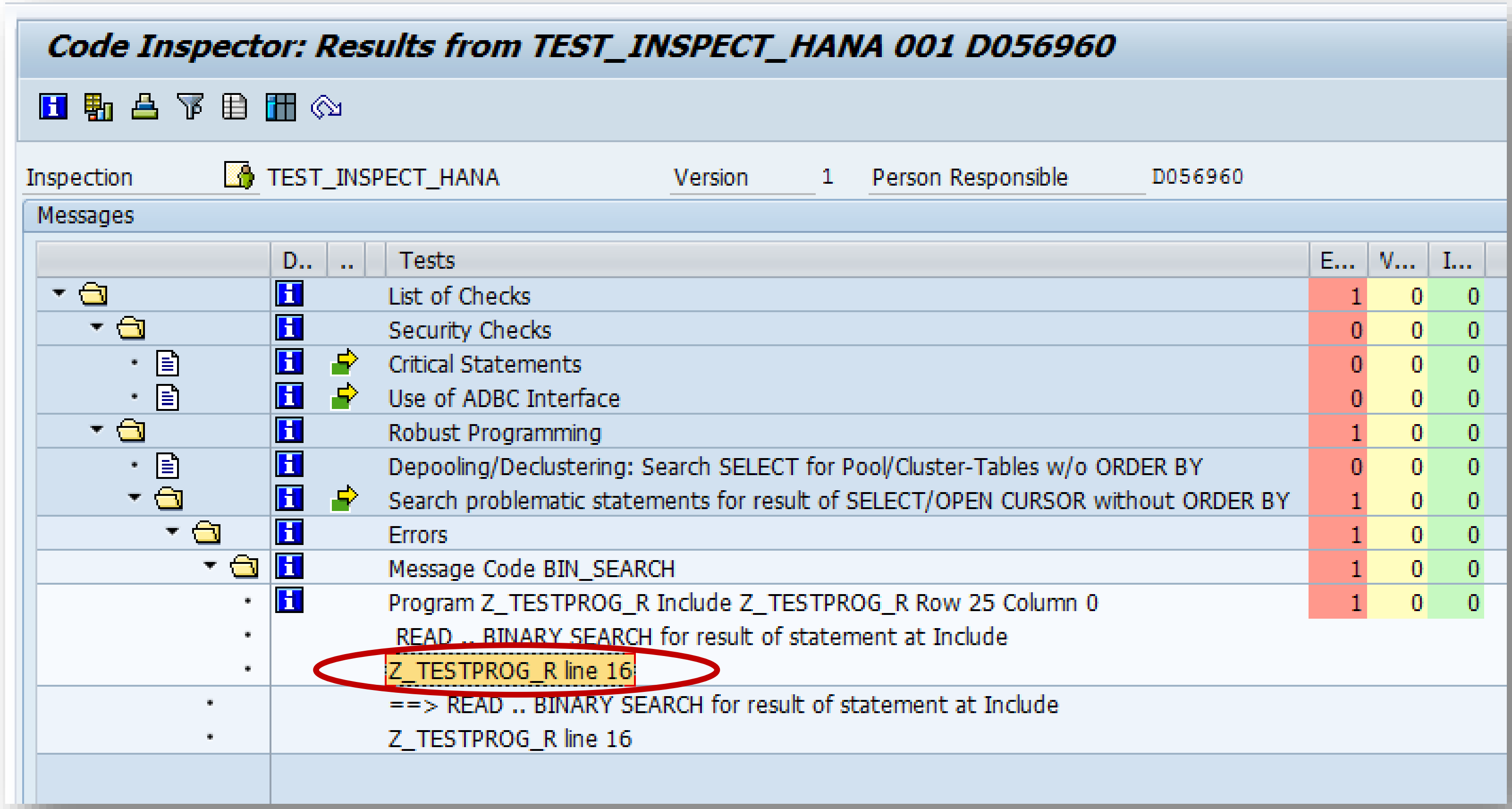
Check Variant

STD_SQL_FOR_HANA

✓ The inspection was carried out successfully

Code Inspector – Step By Step

Step 14: Have a look at the errors found during the inspection. Double-click on the effected objects to see further details.



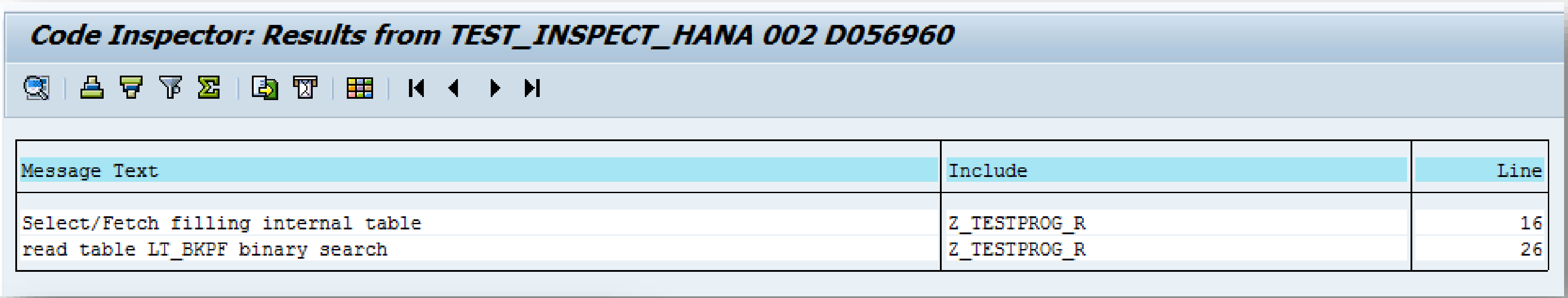
Code Inspector: Results from TEST_INSPECT_HANA 001 D056960

Inspection TEST_INSPECT_HANA Version 1 Person Responsible D056960

Messages

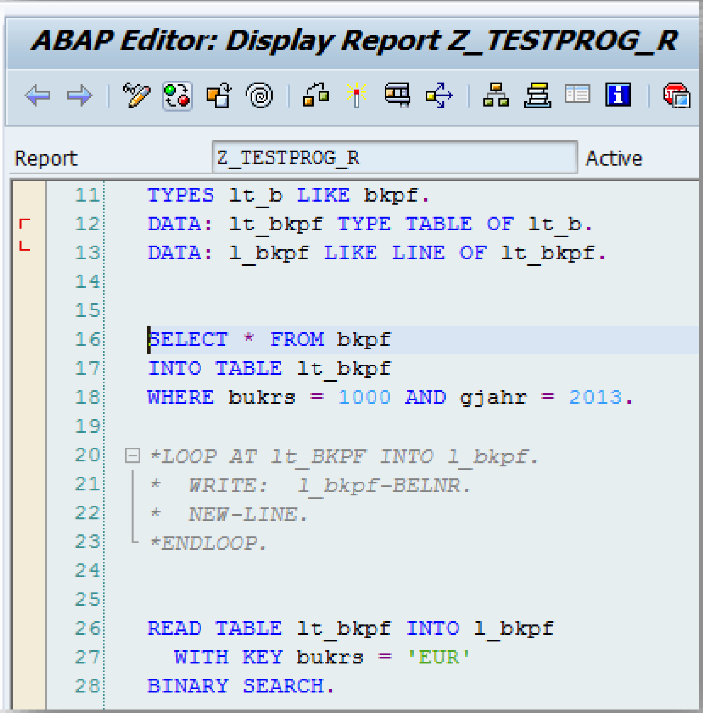
Tests	E...	V...	I...
List of Checks	1	0	0
Security Checks	0	0	0
Critical Statements	0	0	0
Use of ADBC Interface	0	0	0
Robust Programming	1	0	0
Depooling/Declustering: Search SELECT for Pool/Cluster-Tables w/o ORDER BY	0	0	0
Search problematic statements for result of SELECT/OPEN CURSOR without ORDER BY	1	0	0
Errors	1	0	0
Message Code BIN_SEARCH	1	0	0
Program Z_TESTPROG_R Include Z_TESTPROG_R Row 25 Column 0	1	0	0
READ .. BINARY SEARCH for result of statement at Include			
Z_TESTPROG_R line 16			
==> READ .. BINARY SEARCH for result of statement at Include			
Z_TESTPROG_R line 16			

Code Inspector – Step By Step



Code Inspector: Results from TEST_INSPECT_HANA 002 D056960

Message Text	Include	Line
Select/Fetch filling internal table	Z_TESTPROG_R	16
read table LT_BKPF binary search	Z_TESTPROG_R	26



ABAP Editor: Display Report Z_TESTPROG_R

Report Z_TESTPROG_R Active

```
TYPES lt_b LIKE bkpf.
DATA: lt_bkpf TYPE TABLE OF lt_b.
DATA: l_bkpf LIKE LINE OF lt_bkpf.


SELECT * FROM bkpf
INTO TABLE lt_bkpf
WHERE bukrs = 1000 AND gjahr = 2013.

*LOOP AT lt_BKPF INTO l_bkpf.
*   WRITE:   l_bkpf-BELNR.
*   NEW-LINE.
*ENDLOOP.


READ TABLE lt_bkpf INTO l_bkpf
  WITH KEY bukrs = 'EUR'
BINARY SEARCH.
```

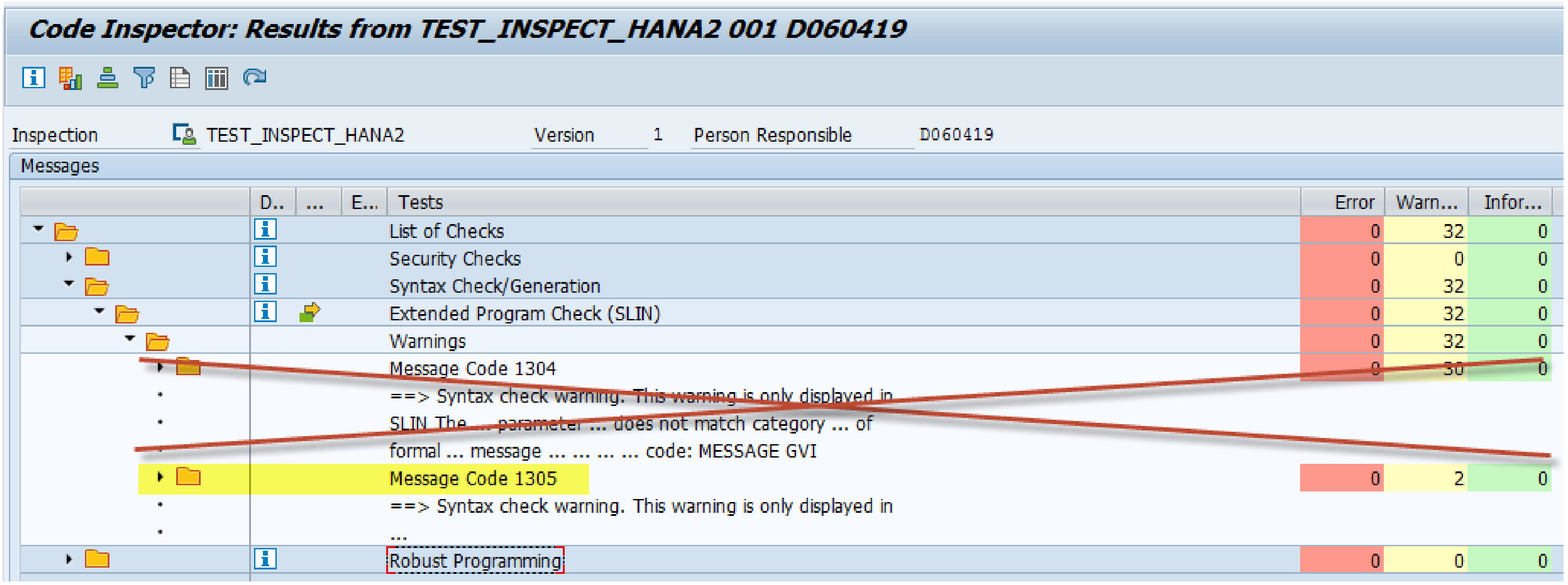
Step 15: Analyze the messages. In this case there is a “binary search” statement which relies on sorted results but no explicit sorting was done. This should be changed. Otherwise it may cause a functional error (see slide “SELECT Statements in HANA – Example”).

Step 16: Double-click on the effected objects to directly navigate into the code and solve the issue.

Code Inspector – Results Explanation

(ONLY NEEDED if SAP NOTE 2251947 is NOT implemented)
If above SAP Note is implemented, you do not have these message and this slide can be ignored!

Warning: Due to the selected “Extended Program Check (SLIN)” check there may be a lot of warnings in the result. Only a fraction of these warnings may be important to make your code compliant for HANA. Only the yellow highlighted messages inside folder “Message Code 1305” may be relevant for HANA Readiness.



Code Inspector: Results from TEST_INSPECT_HANA2 001 D060419

Inspection	TEST_INSPECT_HANA2	Version	1	Person Responsible	D060419

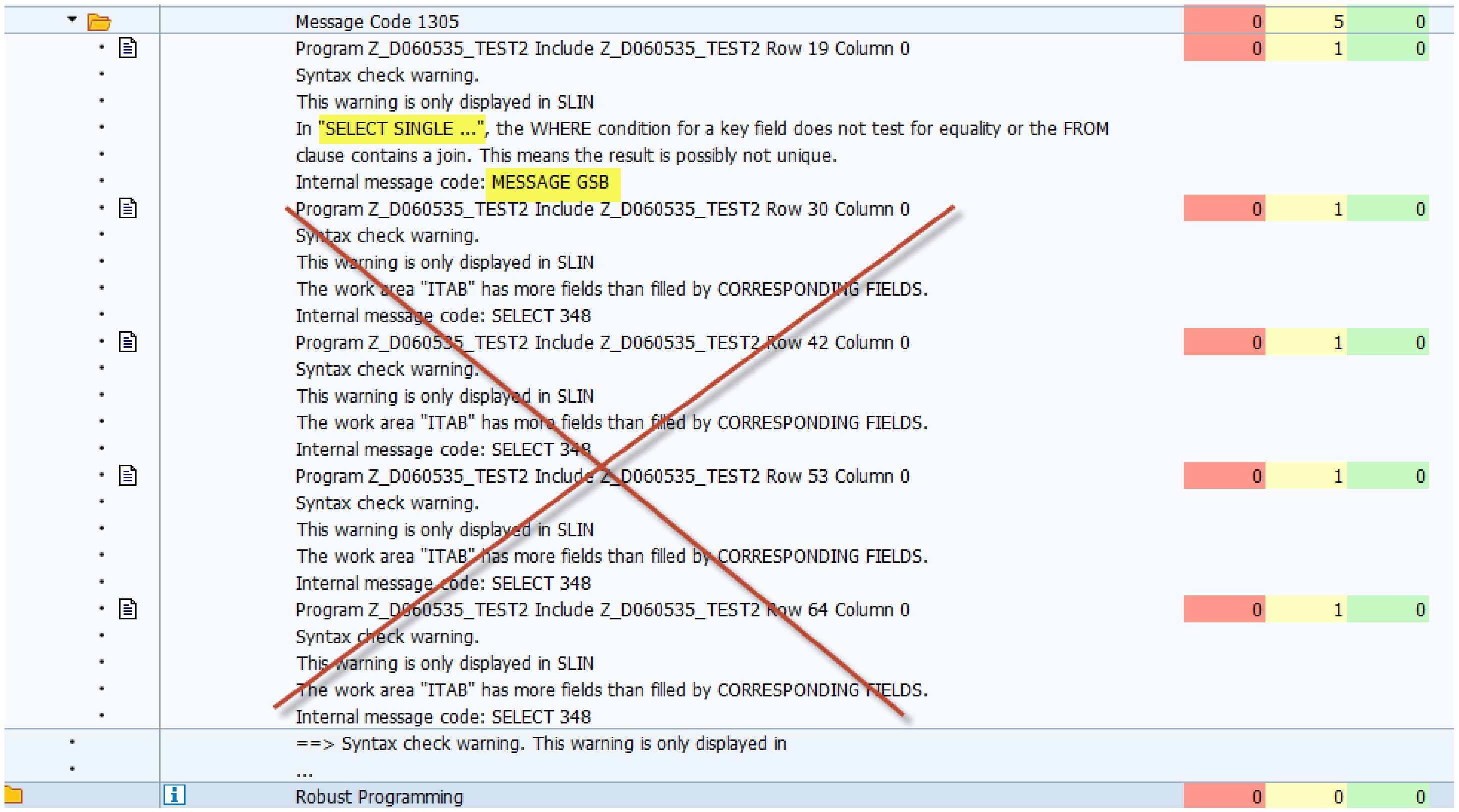
Messages

D..	...	E...	Tests	Error	Warn...	Infor...
			List of Checks	0	32	0
			Security Checks	0	0	0
			Syntax Check/Generation	0	32	0
			Extended Program Check (SLIN)	0	32	0
			Warnings	0	32	0
			~~Message Code 1304~~	~~0~~	~~30~~	~~0~~
			~~==> Syntax check warning. This warning is only displayed in~~			
			~~SLIN The ... parameter ... does not match category ... of~~			
			~~formal ... message ... ... code: MESSAGE GVI~~			
			Message Code 1305	0	2	0
			==> Syntax check warning. This warning is only displayed in			
			...			
			Robust Programming	0	0	0

Code Inspector – Results Explanation

(This slide is ONLY RELEVANT if SAP NOTE 2251947 is NOT implemented)

Inside Folder "Message Code 1305" only items with internal message code "MESSAGE GSB" are relevant.

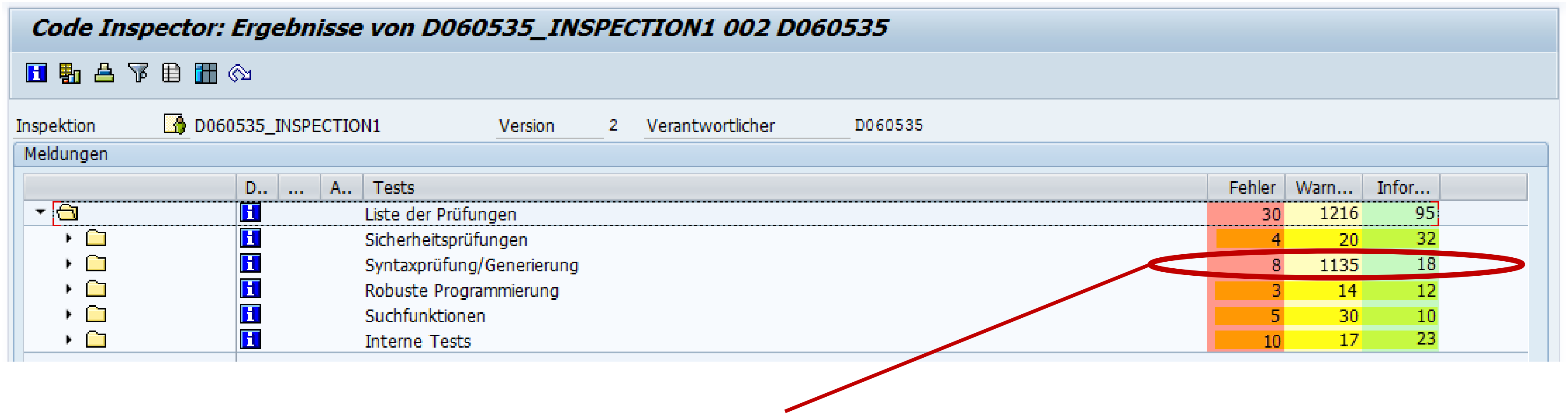


```
Message Code 1305                                                          0    5    0
Program Z_D060535_TEST2 Include Z_D060535_TEST2 Row 19 Column 0            0    1    0
Syntax check warning.
This warning is only displayed in SLIN
In "SELECT SINGLE ...", the WHERE condition for a key field does not test for equality or the FROM
clause contains a join. This means the result is possibly not unique.
Internal message code: MESSAGE GSB
Program Z_D060535_TEST2 Include Z_D060535_TEST2 Row 30 Column 0            0    1    0
Syntax check warning.
This warning is only displayed in SLIN
The work area "ITAB" has more fields than filled by CORRESPONDING FIELDS.
Internal message code: SELECT 348
Program Z_D060535_TEST2 Include Z_D060535_TEST2 Row 42 Column 0            0    1    0
Syntax check warning.
This warning is only displayed in SLIN
The work area "ITAB" has more fields than filled by CORRESPONDING FIELDS.
Internal message code: SELECT 348
Program Z_D060535_TEST2 Include Z_D060535_TEST2 Row 53 Column 0            0    1    0
Syntax check warning.
This warning is only displayed in SLIN
The work area "ITAB" has more fields than filled by CORRESPONDING FIELDS.
Internal message code: SELECT 348
Program Z_D060535_TEST2 Include Z_D060535_TEST2 Row 64 Column 0            0    1    0
Syntax check warning.
This warning is only displayed in SLIN
The work area "ITAB" has more fields than filled by CORRESPONDING FIELDS.
Internal message code: SELECT 348
==> Syntax check warning. This warning is only displayed in
...
Robust Programming                                                         0    0    0
```

Code Inspector – Cost Assessment

(This slide is ONLY relevant if SAP NOTE 2251947 is NOT implemented)

To get a quick cost assessment for your code you should remember that the "Extended Program Check" check shows you a lot of Errors/Warnings/Information which are not important for HANA Readiness.



Code Inspector: Ergebnisse von D060535_INSPECTION1 002 D060535

Inspektion D060535_INSPECTION1 Version 2 Verantwortlicher D060535

Meldungen

D..	...	A..	Tests	Fehler	Warn...	Infor...
			Liste der Prüfungen	30	1216	95
			Sicherheitsprüfungen	4	20	32
			Syntaxprüfung/Generierung	8	1135	18
			Robuste Programmierung	3	14	12
			Suchfunktionen	5	30	10
			Interne Tests	10	17	23

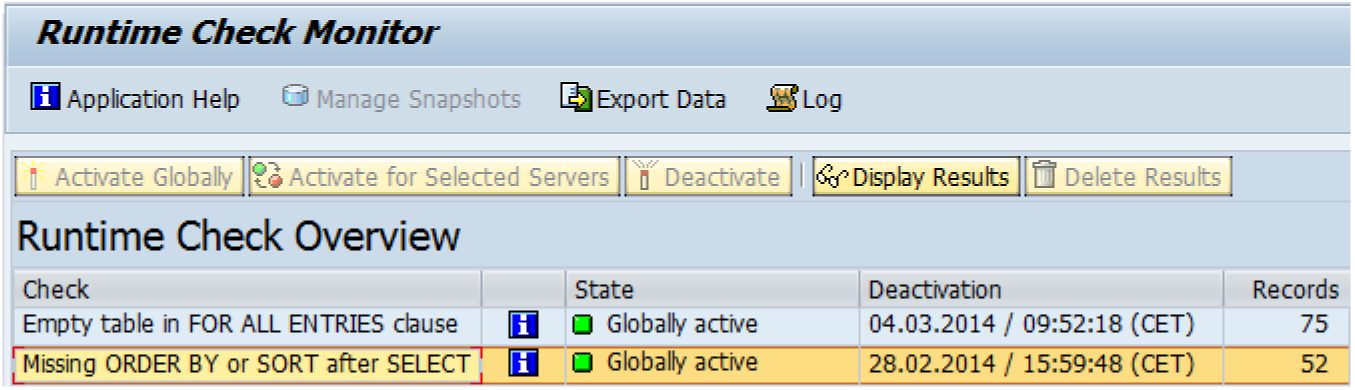
In this example you leave out the "Extended Program Check" because there may be only e.g. 15 messages which are important i.e. SELECT SINGLE messages.

Test Support

As static test have limits a new tool helps to find during runtime if there are any potential functional issue:

Runtime Check Monitor – transaction srtcm

Start srtcm BEFORE your tests and activate the check:
Missing ORDER BY or SORT after SELECT



Runtime Check Overview

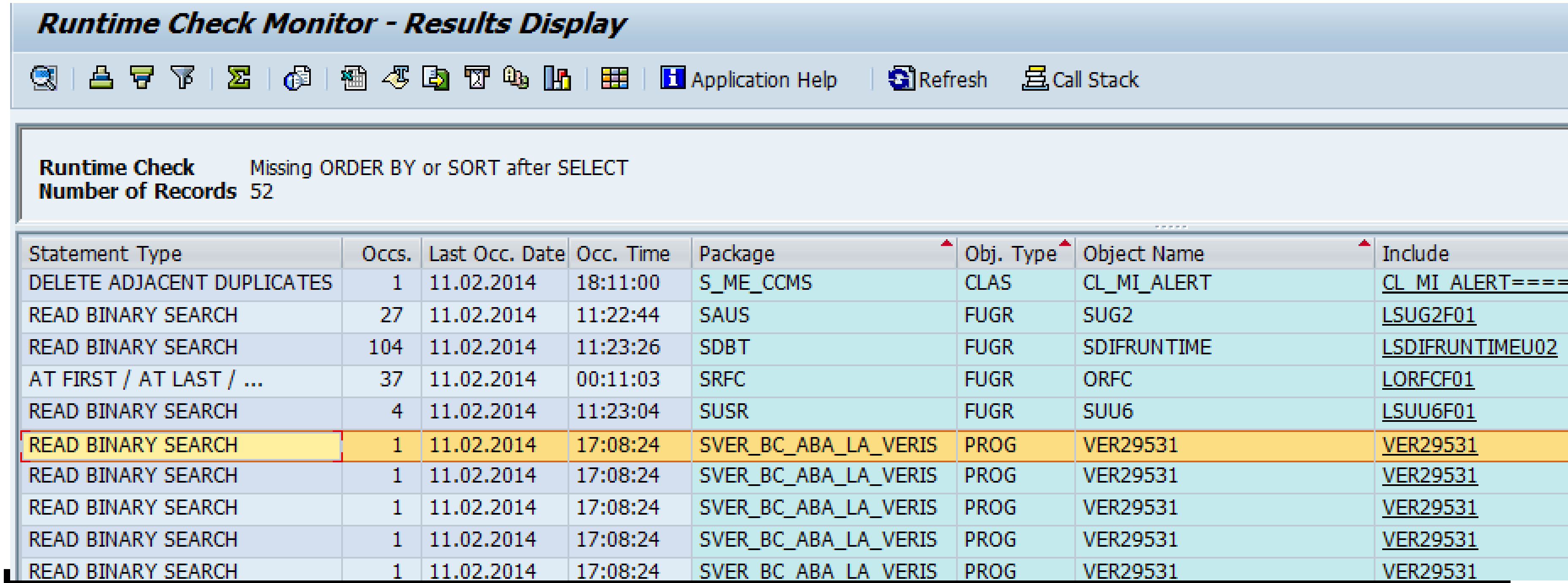
Check		State	Deactivation	Records
Empty table in FOR ALL ENTRIES clause	ℹ	🟩 Globally active	04.03.2014 / 09:52:18 (CET)	75
Missing ORDER BY or SORT after SELECT	ℹ	🟩 Globally active	28.02.2014 / 15:59:48 (CET)	52

After the tests Display Results and evaluate the results

Results of the Runtime Check Monitor

The results of the Runtime Check Monitor show you in each line
The SELECT statement and the corresponding ABAP statement, that needs a sorted table. It only check if a table was sorted by either ORDER BY or SORT.

Note: The Runtime Check Monitor ONLY find issues in coding that is executed, other coding is not seen by this tool.



Runtime Check Missing ORDER BY or SORT after SELECT
Number of Records 52

Statement Type	Occs.	Last Occ. Date	Occ. Time	Package	Obj. Type	Object Name	Include
DELETE ADJACENT DUPLICATES	1	11.02.2014	18:11:00	S_ME_CCMS	CLAS	CL_MI_ALERT	CL_MI_ALERT====
READ BINARY SEARCH	27	11.02.2014	11:22:44	SAUS	FUGR	SUG2	LSUG2F01
READ BINARY SEARCH	104	11.02.2014	11:23:26	SDBT	FUGR	SDIFRUNTIME	LSDIFRUNTIMEU02
AT FIRST / AT LAST / ...	37	11.02.2014	00:11:03	SRFC	FUGR	ORFC	LORFCF01
READ BINARY SEARCH	4	11.02.2014	11:23:04	SUSR	FUGR	SUU6	LSUU6F01
READ BINARY SEARCH	1	11.02.2014	17:08:24	SVER_BC_ABA_LA_VERIS	PROG	VER29531	VER29531
READ BINARY SEARCH	1	11.02.2014	17:08:24	SVER_BC_ABA_LA_VERIS	PROG	VER29531	VER29531
READ BINARY SEARCH	1	11.02.2014	17:08:24	SVER_BC_ABA_LA_VERIS	PROG	VER29531	VER29531
READ BINARY SEARCH	1	11.02.2014	17:08:24	SVER_BC_ABA_LA_VERIS	PROG	VER29531	VER29531
READ BINARY SEARCH	1	11.02.2014	17:08:24	SVER_BC_ABA_LA_VERIS	PROG	VER29531	VER29531

Availability of Runtime Monitor

Availability
SAP NetWeaver 7.40 SP5 OR Via SAP Note 1931870 for SAP NW 7.40 SP2, 3, 4 (SAP_BASIS 740)

ABAP Code Adoption for HANA can start prior to HANA

Start now with the HANA adoption– don’t wait!

?

How you can start without HANA?

Code Inspector and Runtime Check Monitor don’t need HANA!

!

- Code Inspector checks can be done without HANA
  - Find and fix all problematic issues found by the code inspector
- Run the Runtime Check Monitor during all your test
  - (for HANA or other testing)
- RCM may find more code piece that needs to be adjusted
- Keep it on in all system where you test or even productive system if you can

-> Runtime check monitor is meant to be run in productive system

Runtime Check Monitor and Performance impact

Q: Can the Runtime Check Monitor used in a production environment without impacting the productive performance?

A: Yes, the performance overhead is in general not noticeable for an end-user or has a not a significant/noticable impact on performance for batch jobs. Therefore it can be used in production.

Runtime Check Monitor does not check if a table is sorted, it checks if the appropriate ABAP SORT command or ORDER BY was executed on that table!

So it can even find by this concept issues, that currently not occurring!

This extremely fast independently how big the table is!

Test Your Code On a HANA DB

All checks have limits.

Therefore software testing on HANA is always required before going live.

This presentations is sponsored by:

SAP HANA Partner Engineers in the Department of
SAP Partner Innovation Lifecycle Services
click here to watch our **YouTube** video

AND

SAP Vocational Education Department
Studying at SAP / Ausbildung bei SAP

Click here to get more infos

WE NEED YOUR FEEBACK

- it will help the students that worked on this presentation to get good grades and a permanent job at SAP
- sponsor more presentation like this on other topics

If this presentation was helpful to you, please sent an email with your feedback to Michael.Goedecke@SAP.com

Thank You

If you have any questions, please do not hesitate to contact us. We would be happy to answer them and extend this presentation.

Find further information in the best practice guide

http://scn.sap.com/docs/DOC-46714

Contact information

Michael.Goedecke@SAP.com