Fighting Bad PL/SQL

Philipp Salvisberg

Trivadis makes IT easier.

@phsalvisberg

Our company.

Trivadis is a market leader in IT consulting, system integration, solution engineering and the provision of IT services focusing on ORACLE® and Microsoft technologies
in Switzerland, Germany, Austria and Denmark. We offer our services in the following strategic business fields:

Business Intelligence · Application Development · Infrastructure Engineering · Training

OPERATION

Trivadis Services takes over the interacting operation of your IT systems.

trivadis
makes IT easier.

About Me

- Trivadian since April 2000
    - Senior Principal Consultant, Partner
    - Member of the Board of Directors
    - www.salvis.com/blog
    - @phsalvisberg
- Database centric development with Oracle database
- Fond of DSLs to build full stack solutions efficiently and keep them manageable
- Author of free SQL Developer Extensions PL/SQL Cop, PL/SQL Unwrapper, oddgen and Bitemp Remodeler

trivadis
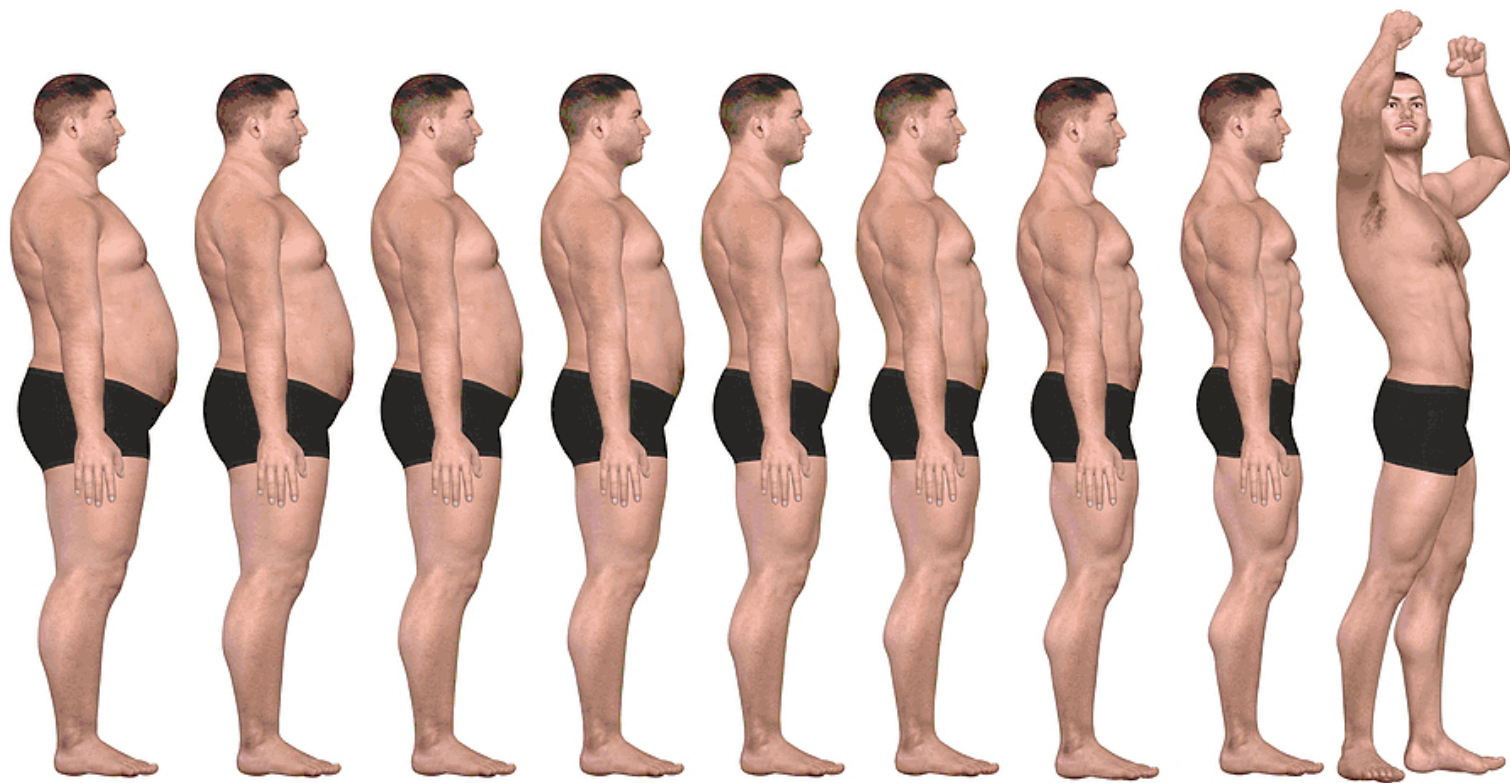makes IT easier.

Agenda

1. **Introduction**

2. **Metrics**

3. **Core messages**

17.11.2016   Fighting Bad PL/SQL

trivadis
makes **IT** easier.

Introduction

trivadis
makes **IT** easier.

Losing Weight

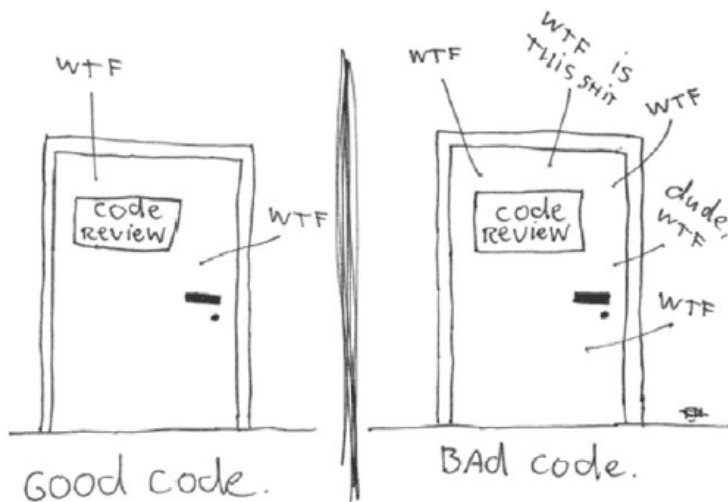trivadis
makes IT easier.

Set Targets – Measure Actuals

- Weight in Kilogram

- Body Fat Percentage

- Skeletal Muscles in Kilogram

- Height in Centimeter

- Abdominal girth in Centimeter

- Girth of … in Centimeter
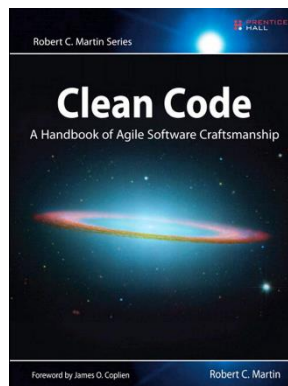- Body-Mass-Index ($bmi = \frac{weightInKg}{heightInMeter^2}$)

trivadis

makes IT easier.

Measuring Code Quality

The ONLY VALID MEASUREMENT OF CODE QUALITY: WTFs/MINUTE

WTF — code review — WTF

Good code.

WTF — WTF is this shit — WTF — code review — dude, WTF — WTF

BAd code.

- "How can we make sure we wind up behind the right door when the going gets tough?"

- "The answer is: *craftsmanship*."

Robert C. Martin Series

PRENTICE HALL

Clean Code

A Handbook of Agile Software Craftsmanship

Foreword by James O. Coplien   Robert C. Martin

Source: http://www.osnews.com/story/19266/WTFs_m; Clean Code, Robert C. Martin, 2009

trivadis
makes IT easier.

Trivadis PL/SQL & SQL Coding Guidelines

Coding Guidelines are a crucial part of software development. It is a matter of fact, that code is more often read than written – therefore we should take efforts to ease the work of the reader, which is not necessarily the author.

I am convinced that this standard may be a good starting point for your own guidelines.

"Roger and his team have done an excellent job of providing a comprehensive set of clear standards that will undoubtedly improve the quality of your code. If you do not yet have standards in place, you should give strong consideration to using these as a starting point."

- Openly available since August 2009
- Download for free from www.trivadis.com

Metrics

17.11.2016   Fighting Bad PL/SQL

trivadis
makes IT easier.

PL/SQL Cop

Checks code against Trivadis PL/SQL & SQL Guidelines. Calculates various metrics.

Command-Line	SonarQube	SQL Developer

▪ Code folder
 	
▪ Snapshot reports
 	
▪ Since 2013
	▪ Code folder
	
	▪ Snapshot/delta reports
	
	▪ Metric repository
	   ▪ Thresholds
	   ▪ Evolvement
	
	▪ Continuous Integration
	
	▪ Since 2015
		▪ Editor content
		
		▪ Snapshot reports
		
		▪ Since 2014 (free)

Download from: https://www.salvis.com/blog/download/

Simple Metrics (Number of …)

```plsql
 1  CREATE OR REPLACE PROCEDURE PASSWORD_CHECK (in_password IN VARCHAR2) IS -- NOSONAR
 2      co_digitarray CONSTANT STRING(10)     := '0123456789';
 3      co_one        CONSTANT SIMPLE_INTEGER := 1;
 4      co_errno      CONSTANT SIMPLE_INTEGER := -20501;
 5      co_errmsg     CONSTANT STRING(100)    := 'Password must contain a digit.';
 6      l_isdigit     BOOLEAN;
 7      l_len_pw      PLS_INTEGER;
 8      l_len_array   PLS_INTEGER;
 9  BEGIN
10      -- initialize variables
11      l_isdigit := FALSE;
12      l_len_pw := LENGTH(in_password);
13      l_len_array := LENGTH(co_digitarray);
14      <<check_digit>>
15      FOR i IN co_one .. l_len_array
16      LOOP
17          <<check_pw_char>>
18          FOR j IN co_one .. l_len_pw
19          LOOP
20              IF SUBSTR(in_password, j, co_one) = SUBSTR(co_digitarray, i, co_one) THEN
21                  l_isdigit := TRUE;
22                  GOTO check_other_things;
23              END IF;
24          END LOOP check_pw_char;
25      END LOOP check_digit;
26      <<check_other_things>>
27      NULL;
28
29      IF NOT l_isdigit THEN
30          raise_application_error(co_errno, co_errmsg);
31      END IF;
32  END password_check;
33  /
```

Metrics

Number of bytes	1,039
Number of lines (LOC)	33
Number of comment lines	1
Number of blank lines	1
Number of net lines	31
Number of commands	1
Number of statements (PL/SQL)	11
Max. cyclomatic complexity	6 ● (● < 11 ⚠ 11..50 ◆ > 50)
Max. Halstead volume	490 ● (● < 1001 ⚠ 1001..3000 ◆ > 3000)
Min. maintainability index (MI)	102 ● (● > 84 ⚠ 64..84 ◆ < 64)
Avg cyclomatic complexity	4 ● (● < 11 ⚠ 11..50 ◆ > 50)
Avg Halstead volume	356 ● (● < 1001 ⚠ 1001..3000 ◆ > 3000)
Avg maintainability index (MI)	104 ● (● > 84 ⚠ 64..84 ◆ < 64)
Number of issues	1
Number of warnings	1
Number of errors	0

PL/SQL Units

PL/SQL Unit	Line	# Lines	# Comment lines	# Blank lines	# Net lines	# Stmts	Cyclomatic complexity	Halstead volume	Maintainability index
PASSWORD_CHECK	9	24	1	1	22	11	6 ●	490 ●	102 ●

Issue Overview

100.0% Guideline 39 violated: Never use GOTO statements in your code.

Issues

Issue#	Line	Type	Message	Code Excerpt
1	22	W	Guideline 39 violated: Never use GOTO statements in your code.	GOTO check_other_things

trivadis
makes IT easier.

McCabe's Cyclomatic Complexity

```
 9  BEGIN
10      -- initialize variables
11      l_isdigit := FALSE;
12      l_len_pw := LENGTH(in_password);
13      l_len_array := LENGTH(co_digitarray);
14      <<check_digit>>
15      FOR i IN co_one .. l_len_array
16      LOOP
17          <<check_pw_char>>
18          FOR j IN co_one .. l_len_pw
19          LOOP
20              IF SUBSTR(in_password, j, co_one) = SUBSTR(co_digitarray, i, co_one) THEN
21                  l_isdigit := TRUE;
22                  GOTO check_other_things;
23              END IF;
24          END LOOP check_pw_char;
25      END LOOP check_digit;
26      <<check_other_things>>
27      NULL;
28
29      IF NOT l_isdigit THEN
30          raise_application_error(co_errno, co_errmsg);
31      END IF;
32  END password_check;
```
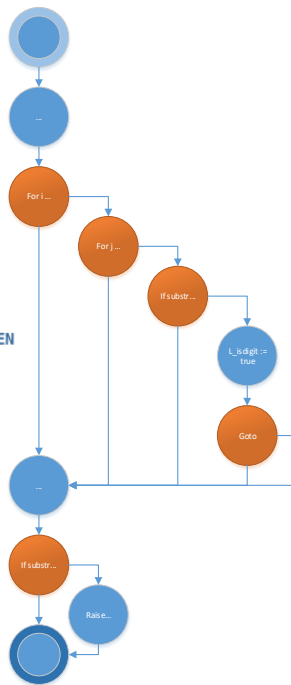
- Number of paths in code
- $M = E - N + 2P$
  - M = Cyclomatic Complexity
  - E = Number of edges
  - N = Number of nodes
  - P = Connected components (number of programs)
- Additional Path for Goto?
  - $15 - 11 + 2*1 = 6$ (Toad)
  - $14 - 11 + 2*1 = 5$ (correct)

Definition: http://www.mccabe.com/pdf/mccabe-nist235r.pdf (1976-1996)

trivadis
makes IT easier.

Cyclomatic Complexity – Drivers & Assessment

- Basic Loops
- Cursor For Loops
- While Loops
- If branches (if, elsif)
- Case branches (when)
- Exception handlers (when)
- Toad Code Analysis compatibility
  - Else in if/case branches
  - PL/SQL blocks
  - Gotos

Cyclomatic Complexity	Complexity evaluation
<11	Reasonable: An average programmer should be able to comprehend and maintain this code.
11..50	Challenging: More senior skills most likely required to comprehend and maintain this code.
>50	Too complex: Candidate for re-design or re-factoring to improve readability and maintainability.

trivadis
makes IT easier.

Halstead Volume

- n1 = number of distinct operators
- n2 = number of distinct operands
- N1 = total number of operators
- N2 = total number of operands
- Program length $N = N1 + N2$
- Program vocabulary $n = n1 + n2$
- Volume $V = N \times \log_2 n$

> Applicable within SQL statements. Plain SQL operators are missing!
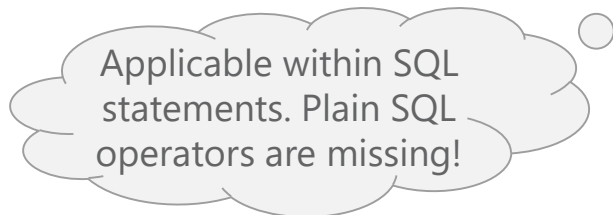
Definition: Elements of Software Science (1977)

- Operators (based on Toad):
  - if, then, elsif, case, when, else, loop, for-loop, forall-loop, while-loop, exit, exit-when, goto, return, close, fetch, open, open-for, open-for-using, pragma, exception, procedure-call, assignment, function-call, sub-block, parenthesis, and, or, not, eq, ne, gt, lt, ge, le, semicolon, comma, colon, dot, like, between, minus, plus, star, slash, percent

- Operands (based on Toad):
  - identifier, string, number

trivadis
makes IT easier.

Halstead Volume – Example

```
 1 CREATE OR REPLACE PROCEDURE PASSWORD_CHECK (in_password IN VARCHAR2) IS -- NOSONAR
 2     co_digitarray CONSTANT STRING(10)      := '0123456789';
 3     co_one        CONSTANT SIMPLE_INTEGER := 1;
 4     co_errno      CONSTANT SIMPLE_INTEGER := -20501;
 5     co_errmsg     CONSTANT STRING(100)     := 'Password must contain a digit.';
 6     l_isdigit     BOOLEAN;
 7     l_len_pw      PLS_INTEGER;
 8     l_len_array   PLS_INTEGER;
 9 BEGIN
10     -- initialize variables
11     l_isdigit := FALSE;
12     l_len_pw := LENGTH(in_password);
13     l_len_array := LENGTH(co_digitarray);
14     <<check_digit>>
15     FOR i IN co_one .. l_len_array
16     LOOP
17         <<check_pw_char>>
18         FOR j IN co_one .. l_len_pw
19         LOOP
20             IF SUBSTR(in_password, j, co_one) = SUBSTR(co_digitarray, i, co_one) THEN
21                 l_isdigit := TRUE;
22                 GOTO check_other_things;
23             END IF;
24         END LOOP check_pw_char;
25     END LOOP check_digit;
26     <<check_other_things>>
27     NULL;
28
29     IF NOT l_isdigit THEN
30         raise_application_error(co_errno, co_errmsg);
31     END IF;
32 END password_check;
33 /
```

- Operators
  - goto: 1, function-call: 4, if: 2, for-loop: 2, comma: 5, not: 1, assignment: 4, semicolon: 19, then: 2, procedure-call: 1, eq: 1

- Operands
  - 'Password must contain a digit.': 1, co_digitarray: 3, check_pw_char: 2, simple_integer: 2, co_errno: 2, raise_application_error: 1, length: 2, false: 1, boolean: 1, check_other_things: 2, substr: 2, 20501: 1, l_len_pw: 3, co_one: 5, l_isdigit: 4, in_password: 2, check_digit: 2, true: 1, j: 2, '0123456789': 1, i: 2, 1: 1, string: 2, pls_integer: 2, co_errmsg: 2, l_len_array: 3

trivadis
makes IT easier.

Halstead Volume – Assessment

- $n1$ = number of distinct operators (11)

- $n2$ = number of distinct operands (26)

- $N1$ = total number of operators (42)

- $N2$ = total number of operands (52)

- Program length $N = N1 + N2$ (94)

- Program vocabulary $n = n1 + n2$ (37)

- Volume $V = N \times \log_2 n$ (490)

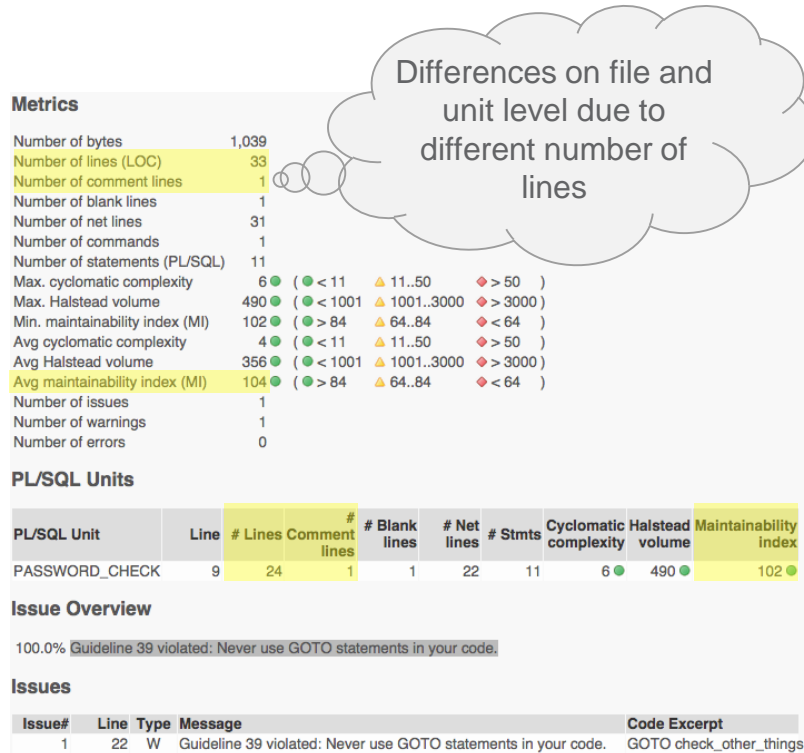| Halstead Volume	Complexity evaluation
<1001	Reasonable: An average programmer should be able to comprehend and maintain this code.
1001..3000	Challenging: More senior skills most likely required to comprehend and maintain this code.
>3000	Too complex: Candidate for re-design or re-factoring to improve readability and maintainability.

trivadis
makes IT easier.

Maintainability Index (MI)

Weighs comments and combines it with Halstead Volume and Cyclomatic Complexity

- $aveV = average\ Halstad\ Volume = \frac{\sum LOCunit \times V}{LOCfile}$

- $aveM = average\ Cyclomatic\ Complexity = \frac{\sum LOCunit \times M}{LOCfile}$

- $aveLOC = average\ lines\ of\ code = \frac{\sum LOCunit}{numberOfUnits}$

- $aveComments = average\ lines\ of\ comment = \frac{\sum linesOfCommentInUnit}{numberOfUnits}$

- $MIwoc = MI\ without\ comments = 171 - 5.2 \times \log_e aveV - 0.23 \times aveM - 16.2 \times \log_e aveLOC$

- $MIcw = MI\ comment\ weight = 50 \times \sin \sqrt{2.4 \times \frac{aveComments}{aveLOC}}$

- $MI = MIwoc + MIcw$

Definition: [The Software Maintainability Index Revisited](#) (1991-2001)

trivadis

makes IT easier.

Maintainability Index (MI) – Assessment

Differences on file and unit level due to different number of lines

MI	Complexity evaluation
>84	Reasonable: An average programmer should be able to comprehend and maintain this code.
64..84	Challenging: More senior skills most likely required to comprehend and maintain this code.
<64	Too complex: Candidate for re-design or re-factoring to improve readability and maintainability.

trivadis
makes IT easier.

```
1 □ CREATE OR REPLACE PROCEDURE PASSWORD_CHECK (in_password IN VARCHAR2) IS
2   BEGIN
3       IF NOT REGEXP_LIKE(in_password, '\d') THEN
4           raise_application_error(-20501, 'Password must contain a digit.');
5       END IF;
6   END;
7   /
```

Metrics

Number of bytes	215
Number of lines (LOC)	7
Number of comment lines	0
Number of blank lines	0
Number of net lines	7
Number of commands	1
Number of statements (PL/SQL)	2
Max. cyclomatic complexity	2 ● (● < 11 △ 11..50 ◆ > 50)
Max. Halstead volume	65 ● (● < 1001 △ 1001..3000 ◆ > 3000)
Min. maintainability index (MI)	123 ● (● > 84 △ 64..84 ◆ < 64)
Avg cyclomatic complexity	1 ● (● < 11 △ 11..50 ◆ > 50)
Avg Halstead volume	46 ● (● < 1001 △ 1001..3000 ◆ > 3000)
Avg maintainability index (MI)	125 ● (● > 84 △ 64..84 ◆ < 64)
Number of issues	5
Number of warnings	5
Number of errors	0

PL/SQL Units

PL/SQL Unit	Line	# Lines	# Comment lines	# Blank lines	# Net lines	# Stmts	Cyclomatic complexity	Halstead volume	Maintainability index
PASSWORD_CHECK	2	5	0	0	5	2	2 ●	65 ●	123 ●

Issue Overview

60.0% Guideline 05 violated: Avoid using literals in your code.
20.0% Guideline 55 violated: Avoid use of the RAISE_APPLICATION_ERROR built-in procedure with a hard-coded - 2
20.0% Guideline 69 violated: Avoid standalone procedures - put your procedures in packages.

Preferences

Trivadis PL/SQL Cop

Guideline check list: [   ]

comma separated list of guidelines to be checked, e.g. "4,12,14,26,30,78"; default = all guidelines

Guideline skip list: [0   ]

comma separated list of guideline checks to be skipped, e.g. "0,5,8"

Honor NOSONAR marker comments: ☑

if checked, do not report guideline violations on lines with NOSONAR marker comments

trivadis
makes IT easier.

Even Better Code?

```
1  CREATE OR REPLACE PROCEDURE PASSWORD_CHECK(in_password IN VARCHAR2)IS BEGIN IF NOT REGEXP_LIKE(in_password,'\d')THEN raise_application_error(-20501,'Password must contain a digit.');END IF;END;
2  /
```

Metrics

Number of bytes	196
Number of lines (LOC)	2
Number of comment lines	0
Number of blank lines	0
Number of net lines	2
Number of commands	1
Number of statements (PL/SQL)	2
Max. cyclomatic complexity	2 ● (● < 11 △ 11..50 ◆ > 50)
Max. Halstead volume	65 ● (● < 1001 △ 1001..3000 ◆ > 3000)
Min. maintainability index (MI)	149 ● (● > 84 △ 64..84 ◆ < 64)
Avg cyclomatic complexity	1 ● (● < 11 △ 11..50 ◆ > 50)
Avg Halstead volume	32 ● (● < 1001 △ 1001..3000 ◆ > 3000)
Avg maintainability index (MI)	153 ● (● > 84 △ 64..84 ◆ < 64)
Number of issues	5
Number of warnings	5
Number of errors	0

WTF?

Add comments for further "improvements"

Improved Maintainability Index by 26!

PL/SQL Units

PL/SQL Unit	Line	# Lines	# Comment lines	# Blank lines	# Net lines	# Stmts	Cyclomatic complexity	Halstead volume	Maintainability index
PASSWORD_CHECK	1	1	0	0	1	2	2 ●	65 ●	149 ●

trivadis
makes IT easier.

Core Messages

trivadis

makes **IT** easier.

Every Metric Has Its Flaws…

■ For example

– *Lines of code* does not account for the code complexity

– *Cyclomatic Complexity* does not account for the length of a program and the complexity of a statement

– *Halstead Volume* does not account for the number of paths in the program

– *Maintainability index* cannot distinguish between useful and useless comments and does not account for code formatting
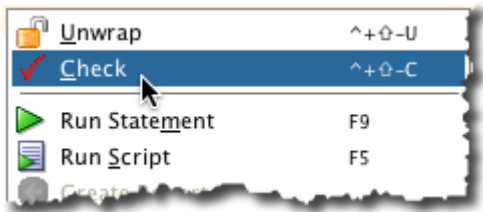

trivadis
makes IT easier.

… But They Are Still Useful

- To Identify complex programs

- To measure code improvements and code degradations

- To help you writing better PL/SQL, if you do not trust in metrics blindly

trivadis
makes IT easier.

Get PL/SQL Cop – Now!

- The PL/SQL Developer extension is free and has no limitations

- Drop me an e-mail if you need an unlimited license key for the command line utility
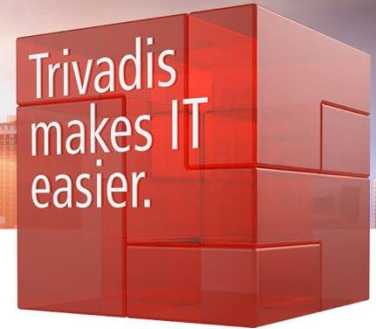

Download from: https://www.salvis.com/blog/download/

trivadis
makes IT easier.

Questions and answers...

Philipp Salvisberg
Senior Principal Consultant

Tel. +41 58 459 52 31
philipp.salvisberg@trivadis.com

Trivadis
makes IT
easier.

trivadis
makes IT easier.

Trivadis @ DOAG 2016

- Booth: 3rd Floor – next to the escalator
- Know how, T-Shirts, Contest and more
- We look forward to your visit
- Because with Trivadis you always win !