Wissen entscheidet.

Bessere Codequalität mit ora* CODECOP

DOAG

DOAG 2023 Konferenz + Ausstellung

Markus Schulze, 22.11.2023 , Nürnberg

Helaba *Invest*

Markus Schulze

Mainhattan

seit 2007 in Frankfurt am Main bei verschiedenen KAGs beschäftigt, seit 2020 bei der Helaba Invest

Oracle

> 15 Jahre Erfahrung als DB Entwickler

> 7 Jahre Erfahrung mit APEX

Certified Associate

Business Intelligence

Microstrategy, Microsoft SQL Server Reporting Services & SAP BusinessObjects

Und sonst?

Familienvater

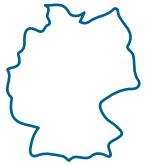
Hobbypianist

Matetrinker

Helaba Invest

#1 S-Finanzgruppe[1]

#4 in Deutschland



223,7 Mrd. Euro

Gesamt-volumen

100% Tochter

der Landesbank Hessen-Thüringen, gegründet 1991

404 Mitarbeitende

am Standort Frankfurt a. M.

Stand: September 2023

[1]Quelle: BVI Investmentstatistik, Marktpositionierung gemessen am Spezialfondsvolumen innerhalb der S-Finanzgruppe

Agenda

Helaba *Invest*

Agenda

Ausgangssituation

Neubau des zentralen Fonds-Datawarehouse

Aus eins mach zwei

/ erforderlich um Wachstum (Skalierbarkeit) und kürzere Entwicklungszyklen zu gewährleisten

/ Wartung des bestehenden und parallel Entwicklung des neuen Datawarehouse

/ strenge Modularisierung (fachlich) sowie Vorrang für Frameworks (technisch)

Produktqualität erfordert Codequalität

/ regulatorische Erfordernisse wie „Einhaltung von Programmierstandards“ oder „Überprüfung des Quellcodes“

/ Ausarbeitung von Programmierrichtlinien als Startpunkt

/ nicht-funktionale Eigenschaften wie Modifizierbarkeit, Anpassbarkeit und Prüfbarkeit verbessern

Agenda

Helaba *Invest*

Statische Codeanalyse

Überschaubares Angebot für PL/SQL: oracle built-in (1)

Compile-time warnings

- Aktivierung auf Sessioneben
  ```
  alter session set plsql_warnings =
  'ENABLE:ALL'
  ```
- view USER|ALL|DBA_ERRORS
  zeigt (Kompilierungs)-Fehler und -Warnungen
- Beispielcodes für Warnungen
  - PLW-06002: unreachable code
  - PLW-06017: an operation will raise an exception
  - PLW-06020: reference to a deprecated entity

PL/Scope

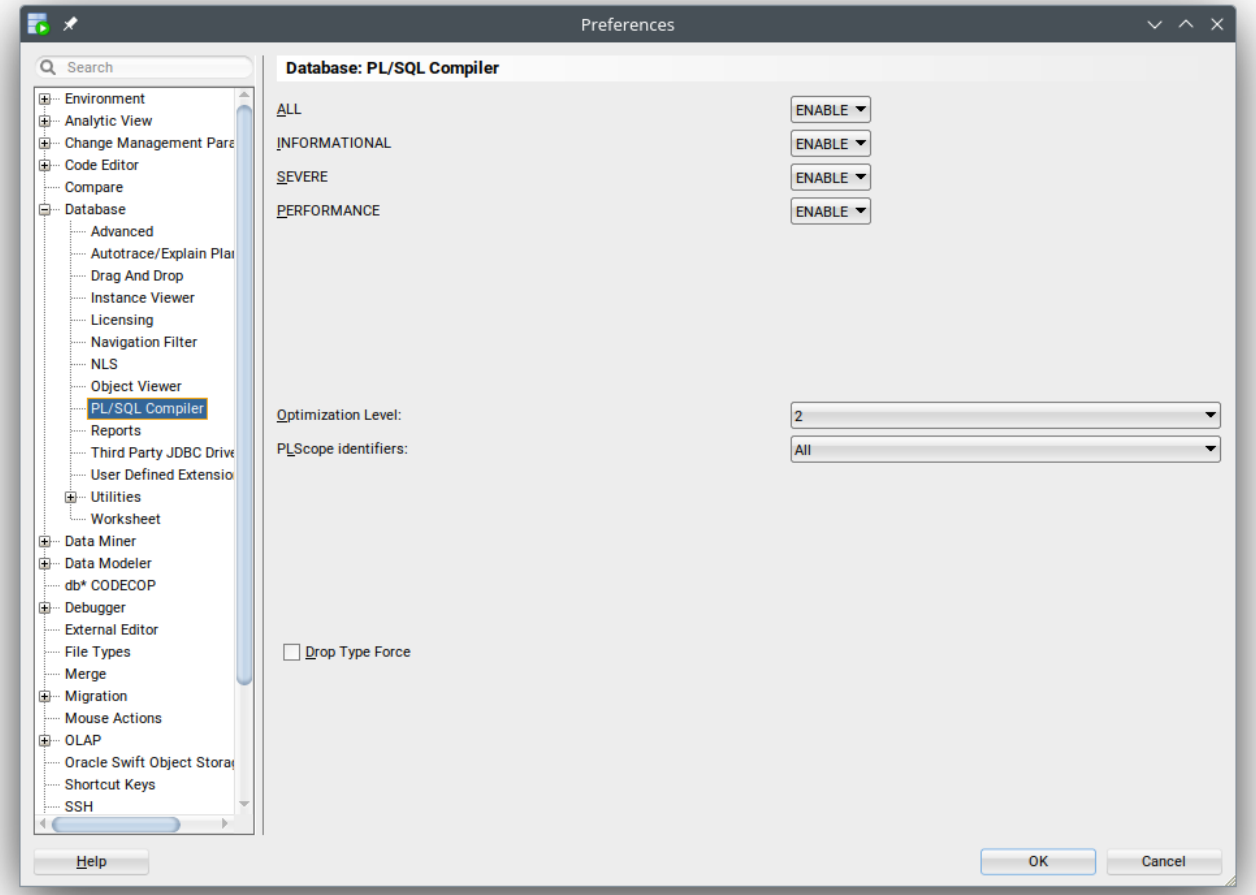
- Aktivierung auf Sessioneben
  ```
  alter session set plscope_settings =
  'IDENTIFIERS:ALL, STATEMENTS:ALL'
  ```
- view USER|ALL|DBA_IDENTIFIERS|STATEMENTS
  zeigt Ergebnisse der Codezerlegung an
- Beispiel-SQL um alle Konstanten im Sourcecode aufzulisten:
  ```
  select *
    from user_identifiers
   where type='CONSTANT'
     and usage='DECLARATION'
  ```

Lint Checkers for PL/SQL by Steven Feuerstein
http://stevenfeuersteinonplsql.blogspot.com/2015/04/lint-checkers-for-plsql.html

Statische Codeanalyse

Überschaubares Angebot für PL/SQL: oracle built-in (2)

Aktivierung in SQL Developer

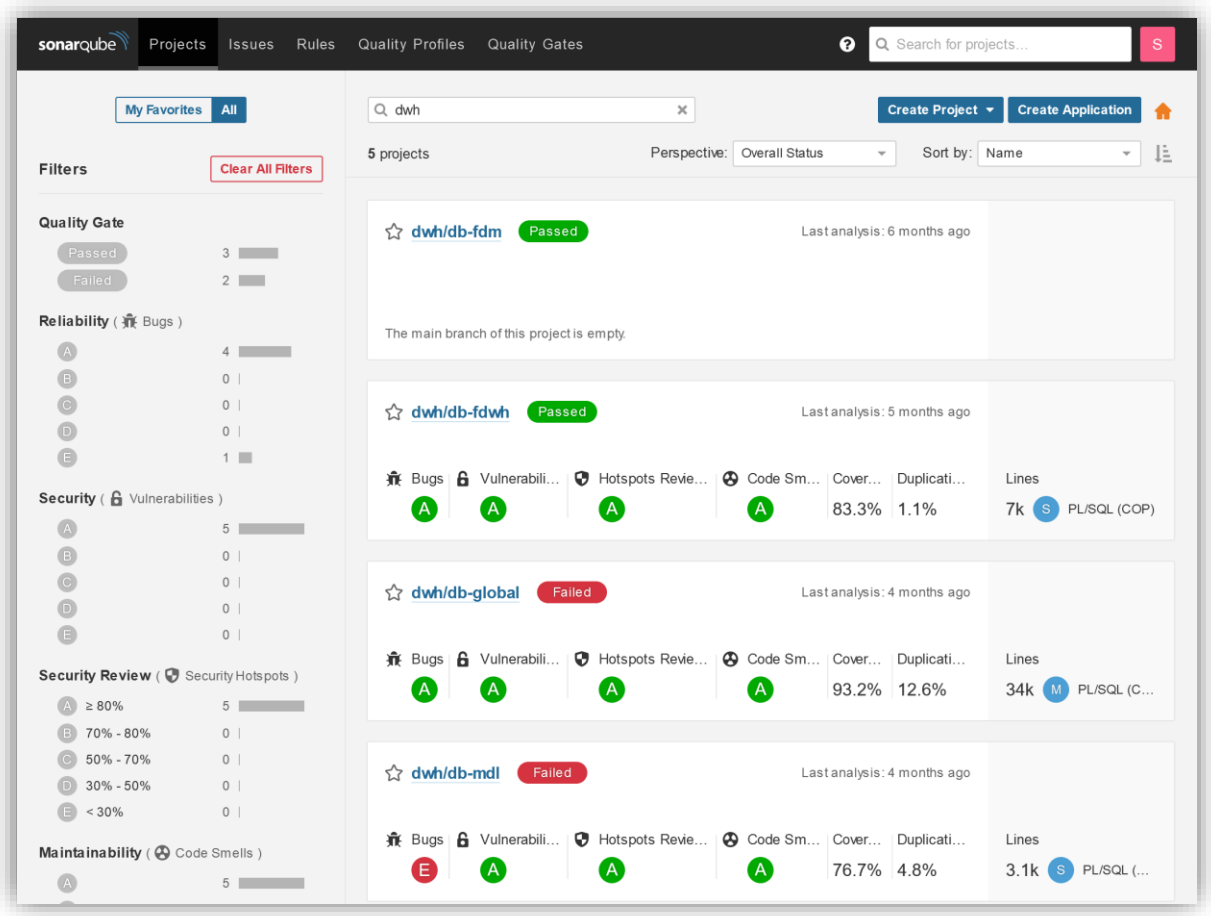


Helaba Invest

Statische Codeanalyse

Überschaubares Angebot für PL/SQL: third-party vendor (1)

SonarQube PL/SQL Analyzer



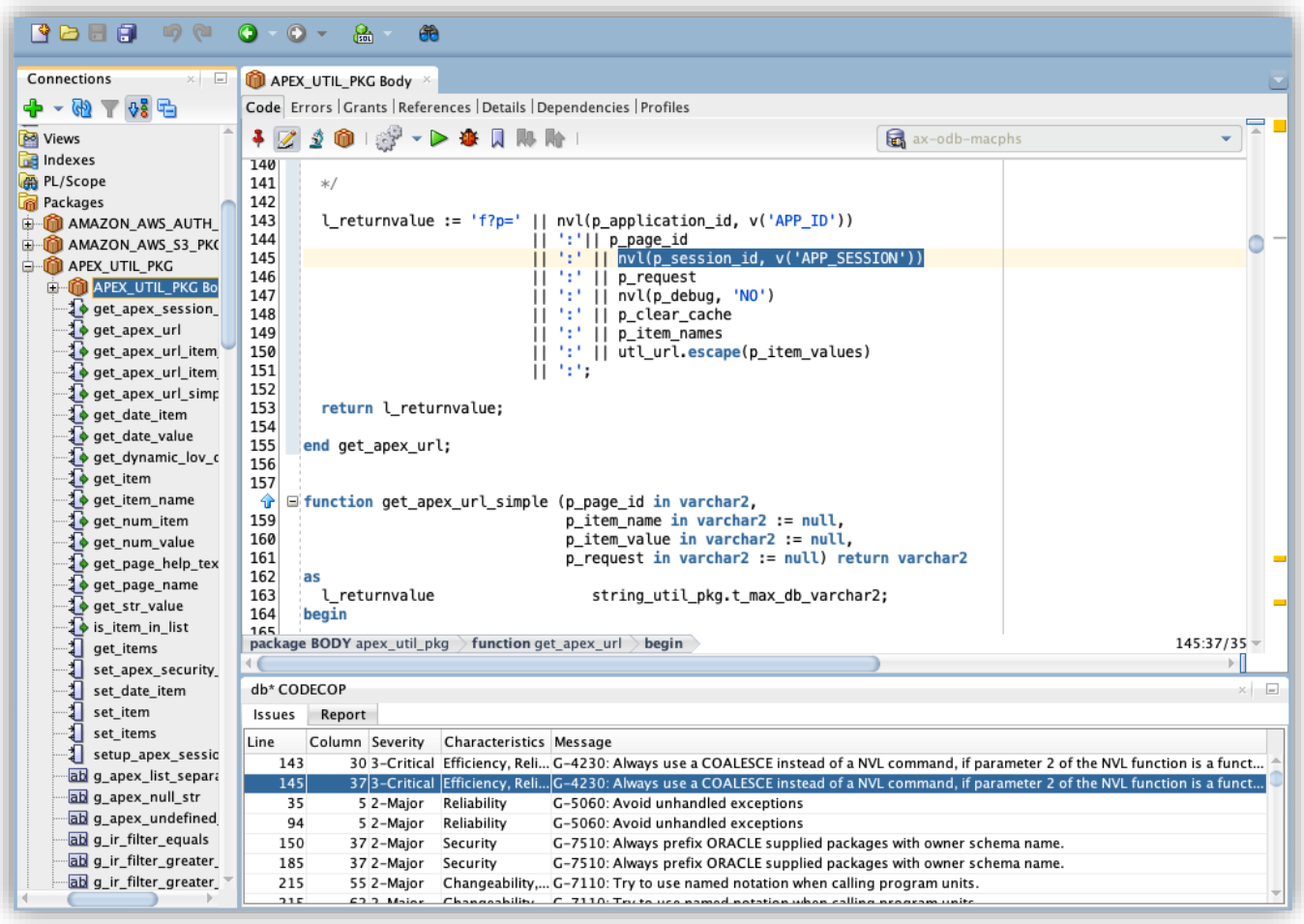
Helaba Invest

Statische Codeanalyse

Überschaubares Angebot für PL/SQL: third-party vendor (2)

db* CODECOP (ehemals PL/SQL Cop)

- prüft Trivadis PL/SQL & SQL Coding Guidelines
- als Plugin für SonarQube und SQL Developer sowie als CLI verfügbar
- Unterschiedliche Kategorisierungen (SQALE characteristics + Severities)



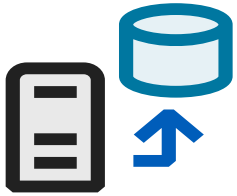
Trivadis PL/SQL & SQL Coding Guidelines
https://github.com/Trivadis/plsql-and-sql-coding-guidelines

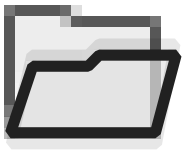
Helaba Invest

Agenda

Deploymentprozess
Schematischer Ablauf

/ Deploymenttool (Eigenentwicklung) → / GIT (gitdiff) → / db* CODECOP (Dateibasiert) - - → / Deploymentordner mit Dateien (Code)





/ …sowie Installationsskript

Visual Studio-Bildbibliothek
https://learn.microsoft.com/de-de/visualstudio/designers/the-visual-studio-image-library?view=vs-2022

Deploymentprozess

Einbindung von hi* CODECOP

Installationsskript

```
connect username1/password@database

prompt /username1/ddl/View
@"username1/ddl/view/myview1.vw"

prompt /username1/ddl/Package
@"username1/ddl/package/mypackage.pkb"

prompt /username1/QualityCheck
hi_codecop.check_schema;

disconnect
...
```

Deploymentprozess

Closed Source vs. Open Source

hi* CODECOP wird ora* CODECOP

- Bewährung als zusätzlicher Baustein der Qualitätssicherung (sinnvolle Erweiterung zu db* CODECOP)
- hi* CODECOP noch ohne Klassifizierungen/Abstufungen von Regeln („alles oder nichts“)
- Wunsch nach „Reparaturknopf“ (bspw. Spaltenkommentare in View aus Tabellen übernehmen)
- bei Ausarbeitung der Prüf-SQLs wurde deutlich das „Data Objects“ genau wie „PL/SQL Units“ oft als ein Bereich betrachtet werden, d.h. unabhängig vom konkreten Objekttypen („alle Spalten“ / „alle Parameter“)
- Idee der Weiterentwicklung bei GitHub als Open Source Projekt und Einreichung Vortrag DOAG 2023
- seit Anfang November online, aktive Mitarbeit erwünscht, insbesondere Entwicklung der „Community Rules“

Agenda

Livedemo

Allgemeiner Überblick und Grundfunktionalität

Berücksichtigung und Prüfung der PL/SQL Compiler Warnungen

Aktivierung dieser neuen Regel in GUI

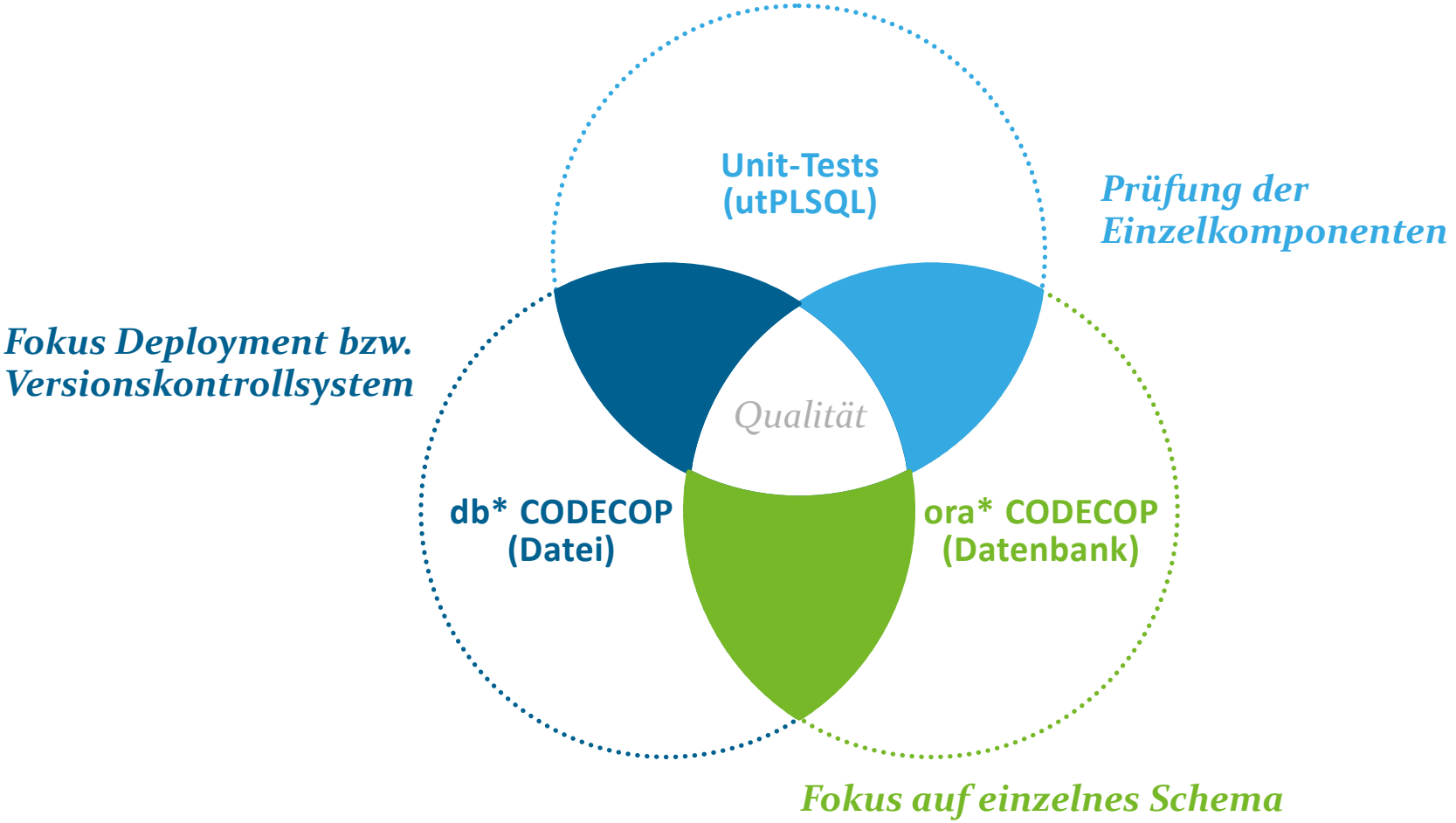
Verifizierung mittels SQL Developer Extension

Transformierung der Regel in einen Unit-Test (utPLSQL)

Agenda

Zusammenfassung

ora* CODECOP als ein zentraler Baustein der Codequalität



Helaba Invest

Ausblick

Potential von ora* CODECOP erst am Anfang

- mit einfachen SQL sehr viel innerhalb der Datenbank prüfbar
- Prüfungen gehen in der Regel - aber nicht nur - auf data dictionary
- teilweise sind Regeln von db* CODECOP ersetzbar
- Transformation als Unittest kann Integration in bestehende Systemlandschaften erleichtern
- APEX GUI erleichtert Wartung und Pflege der Prüfungen
- Aufbau eines Prüfungskataloges durch Community wünschenswert

Vielen Dank für Ihre Aufmerksamkeit!

GitHub

https://github.com/yerba1704/occ

https://github.com/yerba1704/occ-apex

https://github.com/yerba1704/occ-sqldeveloper

https://github.com/yerba1704/occ-utplsql