MT AG

Das QA Framework Quasto

Oliver Lemm

DOAG 21.09.22

über mich

Fachbereichsleiter APEX @ MT AG
Architekt, Projektleiter und Entwickler

ORACLE ACE

ODTUG Kscope

Go DevOps 2019

APEX connect by DOAG

IT-Tage 2019

DOAG Konferenz + Ausstellung

ā'pěks world (#orclapex) (#apexworld)

ā'pěks alpe adria

Agenda

1. what is Quasto?
2. installing Quasto
3. using Quasto
4. utPLSQL Support
5. next Release
6. roadmap
7. other QA tools
8. conclusion

what is Quasto?

QUality
ASsurance
TOol

what is Quasto?

It‘s a PL/SQL Framework to check quality standards from your

database model
pl/sql code
data
apex application

by crawling metadata or table content with SQL Queries

minimum requirements

Oracle Database
no APEX
no utPLSQL
no ci/cd Tool

full feature requirements

Oracle Database 19c
utPLSQL v3*
APEX 22.1*
Jenkins v2*

*full features in coming releases

installing Quasto

Completely free no costs

https://github.com/mt-ag/quasto

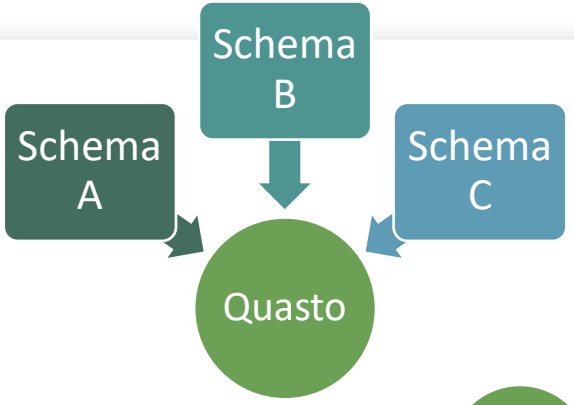
Quasto - Quality Assurance Tool

1. What is Quasto?

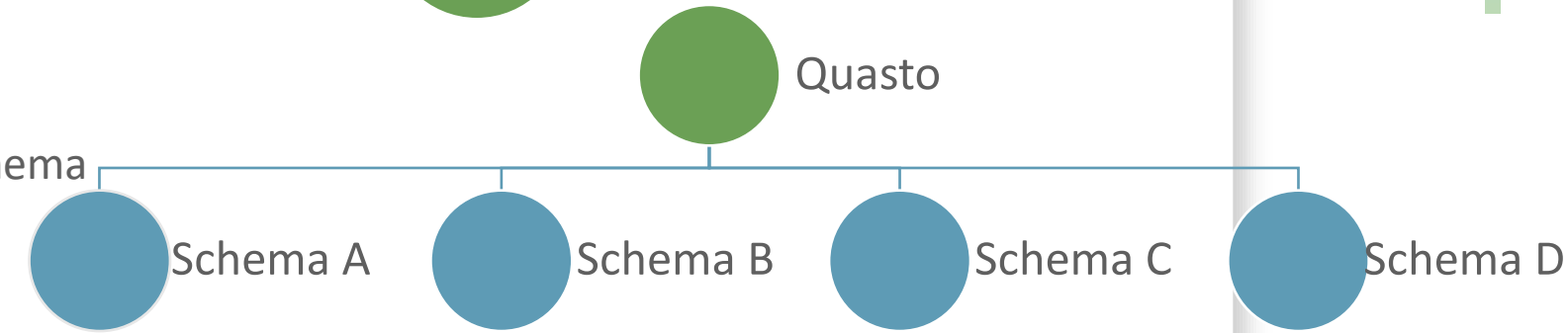
A project for checking guidelines and code quality for inside the oracle database. The first version supports checks for your data model, PL/SQL code and data itself. Coming up releases will integrate utPLSQL and get an APEX Front-End. CI/CD support for jenkins/azure devops or gitlab/github are planned on the roadmap.

installing Quasto – 3 different approaches

1. Quasto is called from other schema



2. Quasto has access on other schema



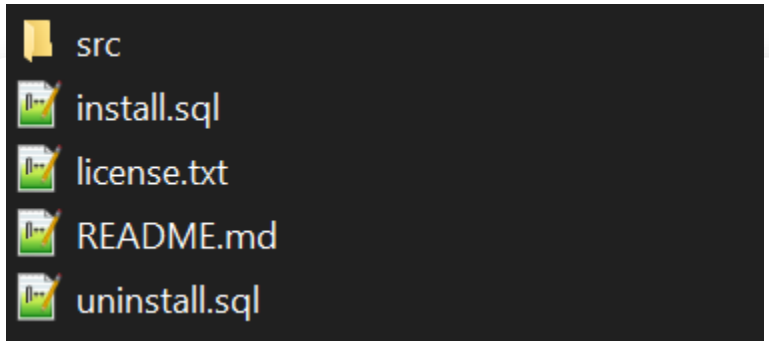
3. Quasto is installed in developing schema

installing Quasto – creating the user/schema

```
grant create procedure to quasto;
grant create public synonym to quasto;
grant create sequence to quasto;
grant create table to quasto;
grant create trigger to quasto;
grant create type to quasto;
grant create view to quasto;

grant create session to quasto;
```

installing Quasto (in schema quasto)




```
C:\Users\olemm\Documents\git_quasto>sql quasto@localhost:1521/orclpdb.mt-ag.com


SQLcl: Release 22.1 Production auf Mo. Aug. 29 11:57:49 2022

Copyright (c) 1982, 2022, Oracle. All rights reserved. Alle Rechte vorbehalten.

Kennwort? (**********?) *******
Verbunden mit:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0

SQL> @install 0 0 0
```

installing Quasto (in schema quasto)

@install 0/1 0/1 0/1

0 = install no utPLSQL supporting objects

0 = install no APEX supporting objects*

0 = install no Jenkins supporting objects*

* will come in later releases

using Quasto

1. Define your internal quality standards
2. Write a rule for every standard
3. Define a set of rules per client/project
4. Run rules and check the results
5. change your objects
6. rerun rules

using Quasto – the rules

Rule Number	=> Unique Identifier together with client name
Client Name	=> Define a set of rules for one client/project
Name	=> Name of your rule
Category	=> DDL, DATA, PLSQL or APEX
Object Types	=> Object type like Table, Trigger…. Colon delimited
Error Message	=> Which message should be shown to the user
Exclude Objects	=> Any objects to not apply to this rule?
Is Active	=> 1=active, 0= not active
SQL	=> Full query which selects all objects

Columns of QA_RULES

table for the rules defined for the QA Tool

	Name	Type	Nu
▶	QARU_ID	NUMBER	
2	QARU_RULE_NUMBER	VARCHAR2(20 CHAR)	
3	QARU_CLIENT_NAME	VARCHAR2(100 CHAR)	
4	QARU_NAME	VARCHAR2(100 CHAR)	
5	QARU_CATEGORY	VARCHAR2(100 CHAR)	
6	QARU_OBJECT_TYPES	VARCHAR2(4000 CHAR)	
7	QARU_ERROR_MESSAGE	VARCHAR2(4000 CHAR)	
8	QARU_COMMENT	VARCHAR2(4000 CHAR)	Y
9	QARU_EXCLUDE_OBJECTS	VARCHAR2(4000 CHAR)	Y
10	QARU_ERROR_LEVEL	NUMBER	
11	QARU_IS_ACTIVE	NUMBER	
12	QARU_SQL	CLOB	
13	QARU_PREDECESSOR_IDS	VARCHAR2(4000 CHAR)	Y
14	QARU_LAYER	VARCHAR2(100 CHAR)	
15	QARU_CREATED_ON	DATE	
16	QARU_CREATED_BY	VARCHAR2(255 CHAR)	
17	QARU_UPDATED_ON	DATE	
18	QARU_UPDATED_BY	VARCHAR2(255 CHAR)	

using Quasto – define a rule (DDL/DML)

Every rule needs to start with set of parameters.
For every check regarding DDL, DML and PLSQL the param WITH-clause should be copied

```
with param as
(select    :1 scheme
         ,:2 qaru_id
       ,:3 qaru_category
      ,:4 qaru_error_level
     ,:5 qaru_object_types
   ,:6 qaru_error_message
        ,:7 qaru_sql
  from dual)
```

```
1  with param as
2   (select :1 scheme
3           ,:2 qaru_id
4           ,:3 qaru_category
5           ,:4 qaru_error_level
6           ,:5 qaru_object_types
7           ,:6 qaru_error_message
8           ,:7 qaru_sql
9    from dual)
10 select qa_rule_t(pi_qaru_id                => p.qaru_id
11                 ,pi_qaru_category          => p.qaru_category
12                 ,pi_qaru_error_level       => p.qaru_error_level
13                 ,pi_qaru_error_message     => p.qaru_error_message
14                 ,pi_qaru_object_types      => p.qaru_object_types
15                 ,pi_qaru_sql               => p.qaru_sql
16                 ,pi_object_id              => null
17                 ,pi_object_name            => at.table_name
18                 ,pi_object_type            => 'TABLE'
19                 ,pi_object_value           => null
20                 ,pi_object_updated_user    => null
21                 ,pi_object_updated_date    => null)
22 from param p
23 cross join all_tables at
24 where (at.owner = p.scheme or p.scheme is null)
25 and (at.owner, at.table_name) not in (select cons.owner
26                                               ,cons.table_name
27                                         from all_constraints cons
28                                         where cons.constraint_type = 'P'
29                                         and cons.status = 'ENABLED')
```

using Quasto – define a rule (DDL/DML)

The resultset has to match the Database Type QA_RULE_T.
For DDL, DML and PL/SQL checks should use this constructor

```
constructor function qa_rule_t
(
  pi_qaru_id              in number
 ,pi_qaru_category        in varchar2
 ,pi_qaru_error_level     in number
 ,pi_qaru_error_message   in varchar2
 ,pi_qaru_object_types    in varchar2
 ,pi_qaru_sql             in clob
 ,pi_object_id            in number
 ,pi_object_name          in varchar2
 ,pi_object_type          in varchar2
 ,pi_object_value         in varchar2
 ,pi_object_updated_user  in varchar2
 ,pi_object_updated_date  in date
) return self as result,
```

```
1  with param as
2   (select :1 scheme
3          ,:2 qaru_id
4          ,:3 qaru_category
5          ,:4 qaru_error_level
6          ,:5 qaru_object_types
7          ,:6 qaru_error_message
8          ,:7 qaru_sql
9    from dual)
10 select qa_rule_t(pi_qaru_id              => p.qaru_id
11                 ,pi_qaru_category        => p.qaru_category
12                 ,pi_qaru_error_level     => p.qaru_error_level
13                 ,pi_qaru_error_message   => p.qaru_error_message
14                 ,pi_qaru_object_types    => p.qaru_object_types
15                 ,pi_qaru_sql             => p.qaru_sql
16                 ,pi_object_id            => null
17                 ,pi_object_name          => at.table_name
18                 ,pi_object_type          => 'TABLE'
19                 ,pi_object_value         => null
20                 ,pi_object_updated_user  => null
21                 ,pi_object_updated_date  => null)
22 from param p
23 cross join all_tables at
24 where (at.owner = p.scheme or p.scheme is null)
25 and (at.owner, at.table_name) not in (select cons.owner
26                                               ,cons.table_name
27                                         from all_constraints cons
28                                         where cons.constraint_type = 'P'
29                                         and cons.status = 'ENABLED')
```

using Quasto – define a rule (DDL/DML)

The Query should bring all Objects related to the parameters you use.

This

```
select *
from all_tables at
where at.owner = 'QUASTO'
and at.table_name not in
  (select ac.table_name
   from all_constraints ac
   where ac.constraint_type = 'P'
   and ac.status = 'ENABLED'
   and ac.owner = 'QUASTO')
```

```
with param as
 (select :1 scheme
        ,:2 qaru_id
        ,:3 qaru_category
        ,:4 qaru_error_level
        ,:5 qaru_object_types
        ,:6 qaru_error_message
        ,:7 qaru_sql
  from dual)
select qa_rule_t(pi_qaru_id              => p.qaru_id
                ,pi_qaru_category        => p.qaru_category
                ,pi_qaru_error_level     => p.qaru_error_level
                ,pi_qaru_error_message   => p.qaru_error_message
                ,pi_qaru_object_types    => p.qaru_object_types
                ,pi_qaru_sql             => p.qaru_sql
                ,pi_object_id            => null
                ,pi_object_name          => at.table_name
                ,pi_object_type          => 'TABLE'
                ,pi_object_value         => null
                ,pi_object_updated_user  => null
                ,pi_object_updated_date  => null)
from param p
cross join all_tables at
where (at.owner = p.scheme or p.scheme is null)
and (at.owner, at.table_name) not in (select cons.owner
                                           ,cons.table_name
                                      from all_constraints cons
                                      where cons.constraint_type = 'P'
                                      and cons.status = 'ENABLED')
```

using quasto – running one rule

```
select *
from qa_api_pkg.tf_run_rule(pi_qaru_rule_number => '23.1'
                           ,pi_qaru_client_name => 'MT AG'
                           ,pi_target_scheme    => 'QUASTO')
```

QARU_CATEGORY	QARU_ERROR_LEVEL	QARU_ERROR_MESSAGE	QARU_OBJECT_TYPES	OBJECT_NAME	OBJECT_TYPE
DDL	1	Table has no primary key defined.	CONSTRAINT	TABELLE_ZUM_TESTEN	TABLE

using quasto – running the rules

```
select *
from qa_api_pkg.tf_run_rules(pi_qaru_client_name => 'MT AG'
                            ,pi_target_scheme    => 'QUASTO')
```

	QARU_ID	QARU_CATEGORY	QARU_ERROR_LEVEL	QARU_ERROR_MESSAGE	QARU_OBJECT_TYPES	QARU_SQL	OBJECT_ID	OBJECT_NAME	OBJECT_TYPE	OBJECT_VALUE
1	26	DDL	1	Table has no primary key defined.	CONSTRAINT	<CLOB>		TABELLE_ZUM_TESTEN	TABLE	
2	29	DDL	1	Foreign key of table has no constraint.	INDEX	<CLOB>		UTPLSQL_TEST_CASE	INDEX OF FK	
3	29	DDL	1	Foreign key of table has no constraint.	INDEX	<CLOB>		UTPLSQL_TEST_SUITE	INDEX OF FK	
4	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368591	TABELLE_ZUM_TESTEN	TABLE COMMENT	
5	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
6	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
7	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
8	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
9	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
10	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
11	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
12	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
13	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
14	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
15	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
16	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
17	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
18	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
19	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
20	32	DDL	1	Table column has no comment.	TABLE	<CLOB>	368492	UTPLSQL_TEST_SUITE	TABLE COMMENT	
21	35	DDL	1	Function is not defined inside a package.	FUNCTION	<CLOB>	368486	FC_EXPORT_QA_RULES	FUNCTION	FC_EXPORT_QA_RULES
22	37	DDL	1	The package body is invalid.	PACKAGE BODY	<CLOB>	368503	PREPARE_TEST_RESULTS_PKG	PACKAGE BODY INVALID	INVALID
23	61	DATA	1	Index name does not correspond to naming convention.	INDEX	<CLOB>	368498	CASE_PK	INDEX	
24	61	DATA	1	Index name does not correspond to naming convention.	INDEX	<CLOB>	368497	CLDR_PK	INDEX	
25	61	DATA	1	Index name does not correspond to naming convention.	INDEX	<CLOB>	368600	QARU_PK	INDEX	
26	61	DATA	1	Index name does not correspond to naming convention.	INDEX	<CLOB>	368499	RUN_PK	INDEX	
27	61	DATA	1	Index name does not correspond to naming convention.	INDEX	<CLOB>	368500	SUITE_PK	INDEX	
28	27	DDL	1	Primary key of table has no comment.	CONSTRAINT	<CLOB>	368492	UTPLSQL_TEST_SUITE	PK COMMENT	
29	27	DDL	1	Primary key of table has no comment.	CONSTRAINT	<CLOB>	368492	UTPLSQL_TEST_SUITE	PK COMMENT	

utPLSQL Support

utPLSQL v3.x is supported

utPLSQL has to be installed seperately

utPLSQL Installation

utPLSQL Install Guide
http://www.utplsql.org/utPLSQL/latest/userguide/install.html

Install Guide by Quasto for utPLSQL
https://github.com/mt-ag/quasto#installing-guidelines-and-specifications-for-the-utplsql-framework

using utPLSQL Integration from Quasto use
@install 1 0 0

why utPLSQL?

XML Files with Test results are standardized

Unit Testing XML Files can be visualized by different tools

utPLSQL is an established framework

how is utPLSQL integrated?

utPLSQL Tests will be generated based on the rules you defined

no utPLSQL Package has to be written by hand

simple procedure call is necessary to generate utPLSQL Packages

additional hand written packages can be combined

why not using utPLSQL without Quasto?

You need to know how to write utPLSQL Tests

Packages has to be written manual

need a single test for every object and not one test for all objects

wanna be faster

less failures

first Release

Quasto 1.0

check DDL, DML, data or APEX apps by your own rules

single or multi scheme support

single or multi project/client support

easy installable framework

next Release

Quasto 1.1

finalize utPLSQL Support

easy export and import for rules

short documentation for CI/CD support

roadmap

APEX Integration as Region Plugin

APEX App to maintain rules

full Integration into CI/CD like Jenkins, Gitlab or Azure DevOps

other Tools

APEX Advisor by Oracle

APEX Project Eye by united codes

SonarQube

utPLSQL by Open Source

QA Plugin by MT AG

conclusion

add a rule for your Oracle DB easier than ever

one rule is a single SQL

utPLSQL integrated without writing utPLSQL Packages manual

Quasto can be used with or without APEX installed

Source Code is Open Source

MT AG

Vorträge Mittwoch, 21.09.2022

Titel	Uhrzeit	Raum	Referent:in
You‘ve got Mail!	15.00–15.45 Uhr	Istanbul	 Timo Herwix
Mobile frei definierbare & skalierbare DevOp’s Umgebung (Docker, Oracle, APEX)	15.00–15.45 Uhr	Kopenhagen	 Guido Sokolis
Web-Server und Apache Tomcat verstehen	16.00–16.45 Uhr	Kopenhagen	 Robin Schulz
Der „Hype“ Podcast: Von der Idee zur Umsetzung #DevsOnTape	16.00–16.45 Uhr	Budapest	 Kai Donato und Carolin Hagemann
Ein Einstieg in Oracle REST Data Services für autonome Datenbanken	17.00–17.45 Uhr	Kopenhagen	 Timo Herwix

MT AG

Vorträge Donnerstag, 22.09.2022

Titel	Uhrzeit	Raum	Referent:in
Playwright enters the stage - End-to-end Oberflächentests für Oracle APEX Apps	08.00–08.45 Uhr	Prag	 Fabian Neureiter
Immersive Analytics: Datenvisualisierung mit VR	09.00–09.45 Uhr	Oslo	 Philipp Hartenfeller
Die Dynamic Actions Schulbank	09.00–09.45 Uhr	Stockholm	 Robin Schulz
APEX Visualizer 22.1	11.00–11.45 Uhr	Sydney	 Oliver Lemm
APEX APIs for Everyone - Reloaded	13.00–13.45 Uhr	Sydney	 Moritz Klein

Haben wir Ihr Interesse geweckt?

Oliver Lemm
Fachbereichsleiter APEX

Telefon: +49 2102 30 961-164
Mail: oliver.lemm@mt-ag.com

MT AG
Balcke-Dürr-Allee 9
40882 Ratingen

www.mt-ag.com

@OliverLemm

https://www.linkedin.com/in/oliverlemm

https://www.xing.com/profile/Oliver_Lemm