Testing with utPLSQL

Made Easy with SQL Developer

Philipp

@phsalvisberg https://www.salvis.com/blog

Philipp

- Database centric development
- Model Driven Software Development
- Author of free SQL Developer Extensions PL/SQL Unwrapper, PL/SQL Cop, utPLSQL, plscope-utils, oddgen and Bitemp Remodeler

@phsalvisberg

https://www.salvis.com/blog

Agenda

trivadis

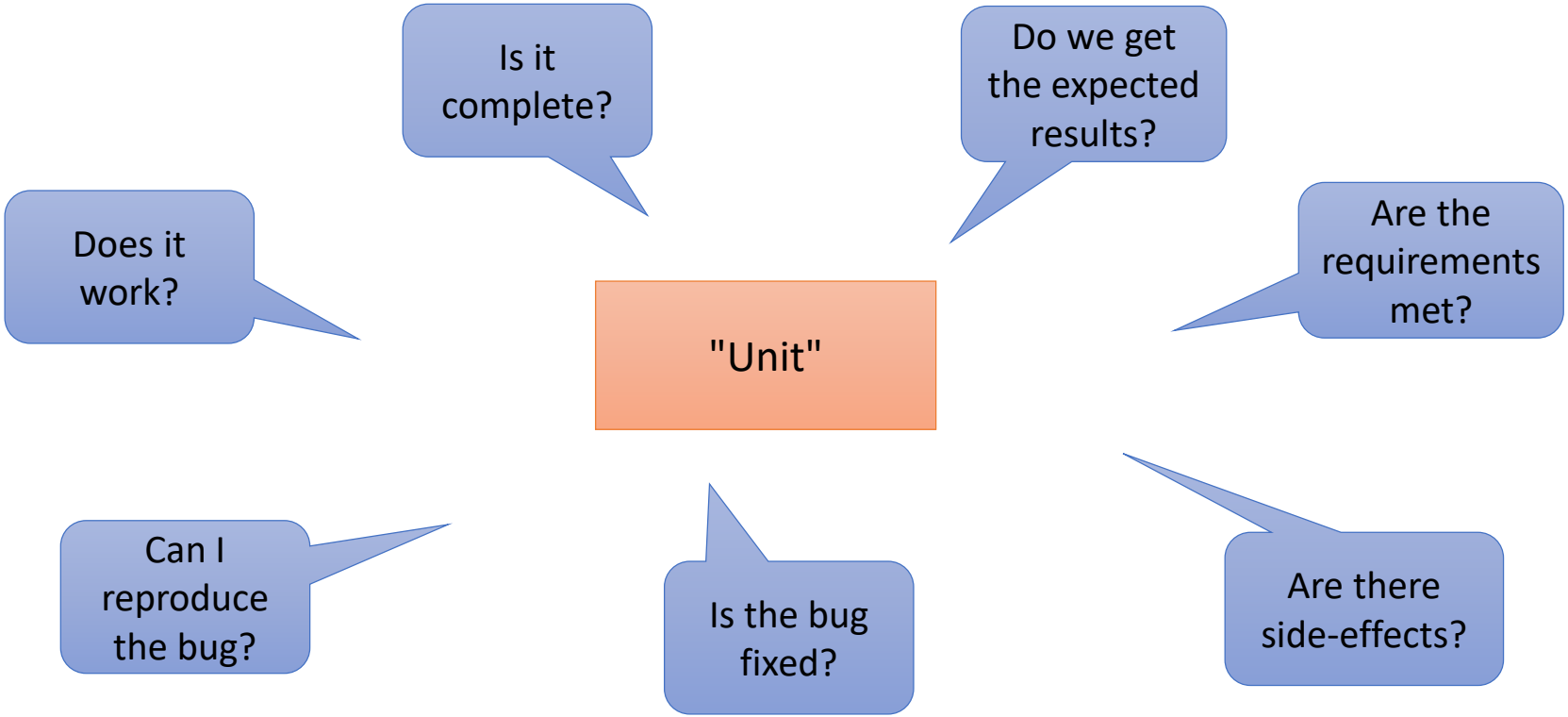
1. Introduction
2. Installation
3. Build & Run Tests in SQL Developer
4. Run Code Coverage Reports in SQL Developer
5. Core Messages

trivadis

Introduction

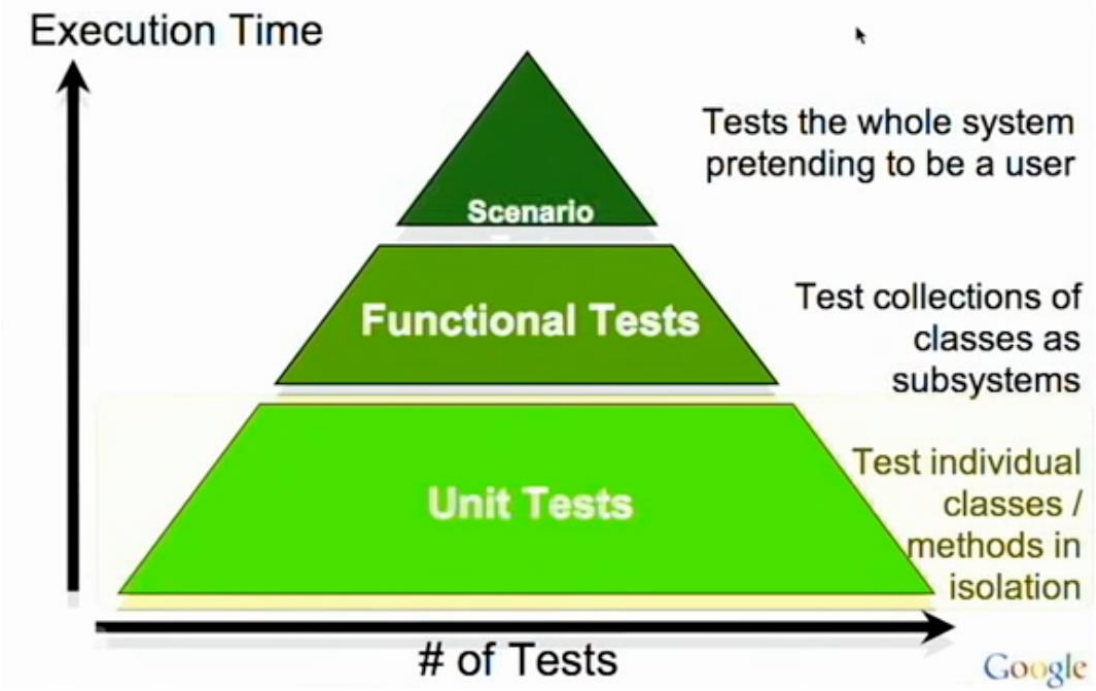
Why?

trivadis



utPLSQL Test Scope

trivadis



Source: Miško Hevery, The Clean Code Talks, Unit Testing, October 30, 2008, https://www.youtube.com/watch?v=wEhu57pih5w&t=991

- GUI

utPLSQL

- API
- Integration
- Components
- Unit

utPLSQL Units Under Test

trivadis

Primary

- Types
- Packages
- Procedures
- Functions

Secondary

- Non-PL/SQL Units
- Views
- Triggers
- Tables

utPLSQL Suite – Apache 2.0 License

trivadis

Mandatory

- Core Testing Framework
  - Schema installed in Oracle DB
  - No repository
  - Annotation based tests

Optional

- Command Line Client
- Maven Plugin
- SQL Developer Extension

ORACLE DATABASE

Maven JUnit TFS TC

Jenkins sonarqube

Test Declaration

trivadis

```
CREATE OR REPLACE PACKAGE test_package_name AS
   --%suite

   --%test
   PROCEDURE procedure_name;
END;
```

--%suite(<description>)
--%suitepath(<path>)
--%tags(<tag>[,...]
--%displayame(<description>)
--%beforeall([...])
--%afterall([...])
--%beforeeach([...])
--%aftereach([...])
--%rollback(manual)
--%disabled
--%context
--%endcontext

--%displayname(<description>)
--%test(<description>)
--%tags(<tag>[,...]
--%throws(<exception>[,...])
--%beforeall
--%afterall
--%beforeeach
--%aftereach
--%beforetest([...])
--%aftertest([...])
--%rollback(manual)
--%disabled

Test Implementation

trivadis

```
CREATE OR REPLACE PACKAGE BODY test_package_name AS
   PROCEDURE procedure_name IS
      l_actual   INTEGER := 0;
      l_expected INTEGER := 1;
   BEGIN
      ut.expect(l_actual).to_equal(l_expected);
   END procedure_name;
END;
```

Matcher:

be_between, be_empty, be_false, be_greater_than, be_greater_or_equal, be_less_or_equal, be_less_than, be_like, be_not_null, be_null, be_true, equal, have_count, match

Extended options for refcursor, object type, JSON, nested table and varray:

- include(<items>)
- exclude(<items>)
- unordered
- join_by(<items>)

Test Run

trivadis

```
SET SERVEROUTPUT ON SIZE UNLIMITED
EXEC ut.run('test_package_name')
```

```
test_package_name
  procedure_name [.003 sec] (FAILED - 1)

Failures:

  1) procedure_name
      Actual: 0 (number) was expected to equal: 1 (number)
      at "TEST_PACKAGE_NAME.PROCEDURE_NAME", line 7 ut.expect(l_actual).to_equal(l_expected);


Finished in .007015 seconds
1 tests, 1 failed, 0 errored, 0 disabled, 0 warning(s)
```

trivadis

Installation

Install utPLSQL Core Testing Framework

trivadis

- Download utPLSQL.zip from https://github.com/utPLSQL/utPLSQL/releases
- Unzip utPLSQL.zip
- cd source
- sqlplus / as sysdba @install_headless.sql
  - User UT3
  - Password XNtxj8eEgA6X6b6f
  - Tablespace USERS

Install utPLSQL for SQL Developer

trivadis

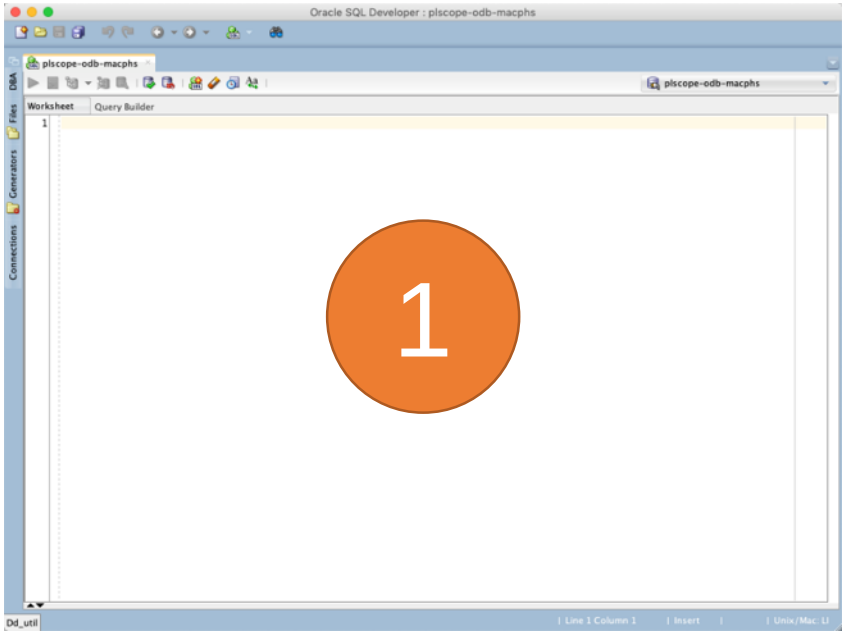
- Download utplsql_for_SQLDev_*.zip
  from https://github.com/utPLSQL/utPLSQL-SQLDeveloper/releases
- Start SQL Developer
- Select "Check for Updates…" in the help menu
- Use the "Install From Local File" option to install the previously downloaded "utplsql_for_SQLDev_*.zip" file
  - User must have read/write access to SQL Developer installation directory
  - Run as Administrator, if required
- Restart SQL Developer

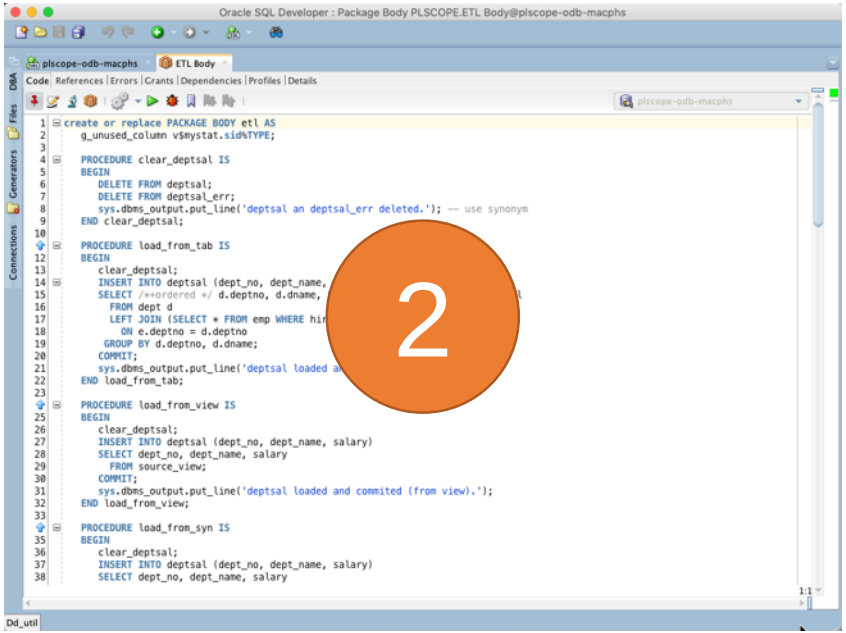
trivadis

Build & Run Tests in SQL Developer

Starting Point?

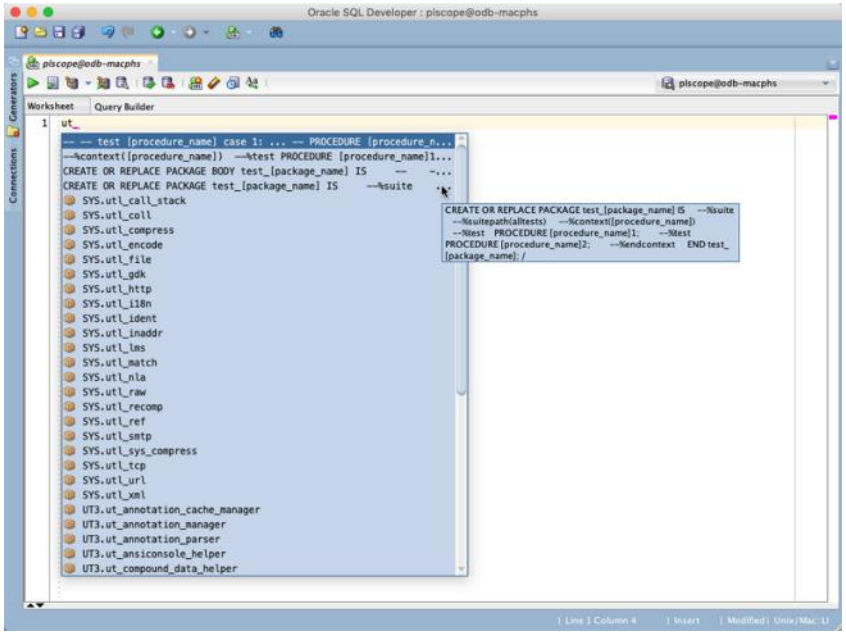
trivadis

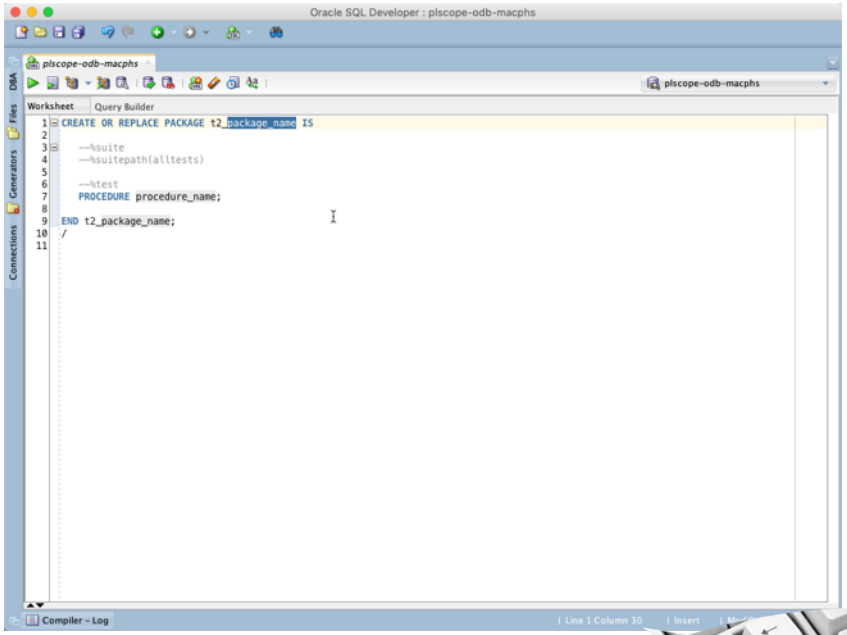




Test First – Create Test from Template

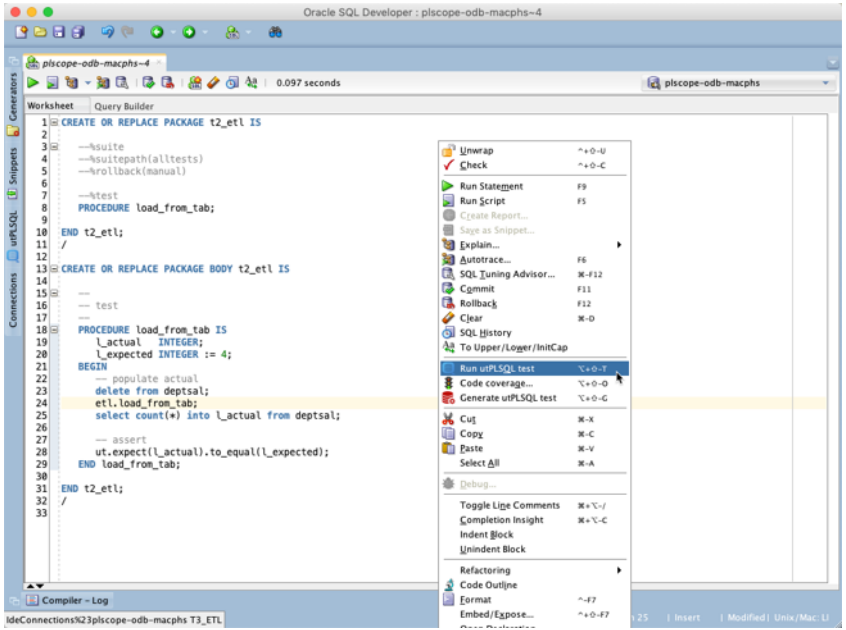
trivadis

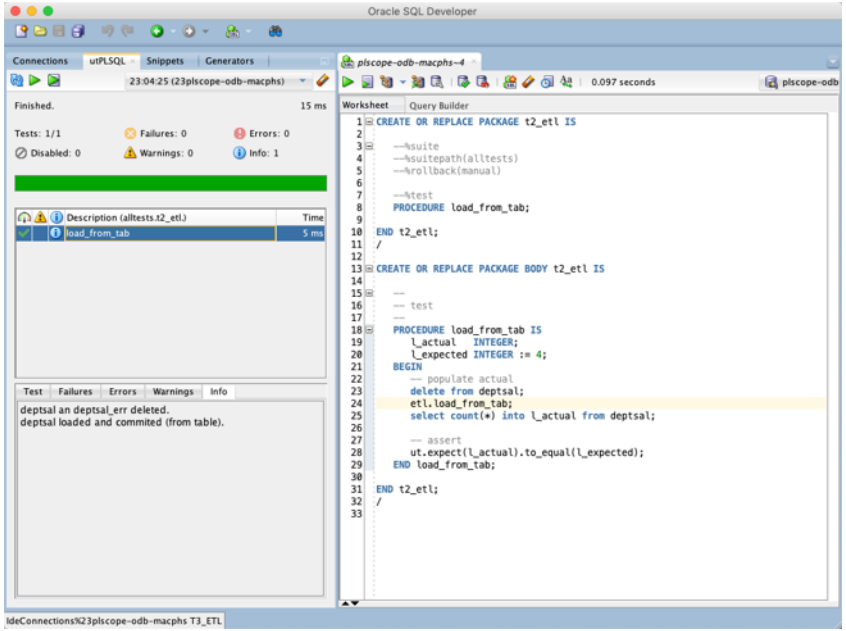




Test First – Complete Test & Run

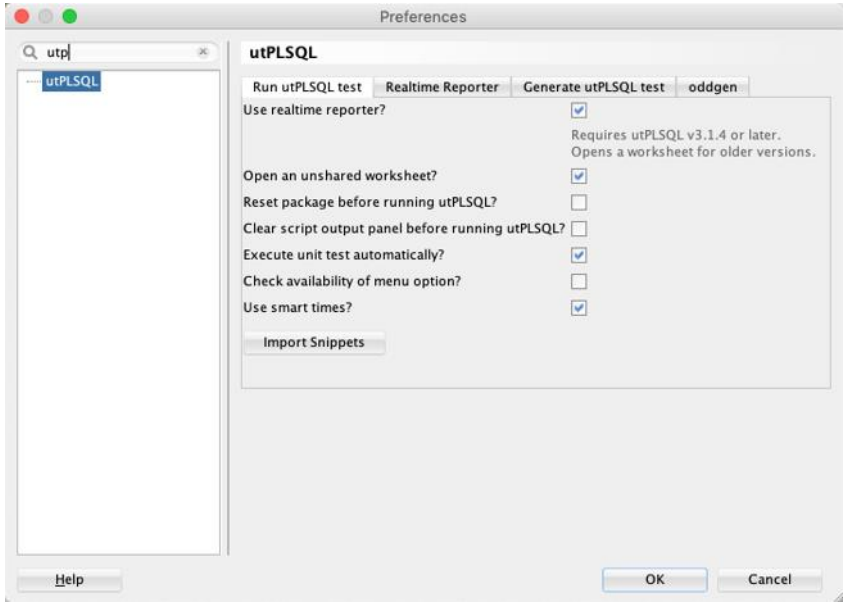
trivadis

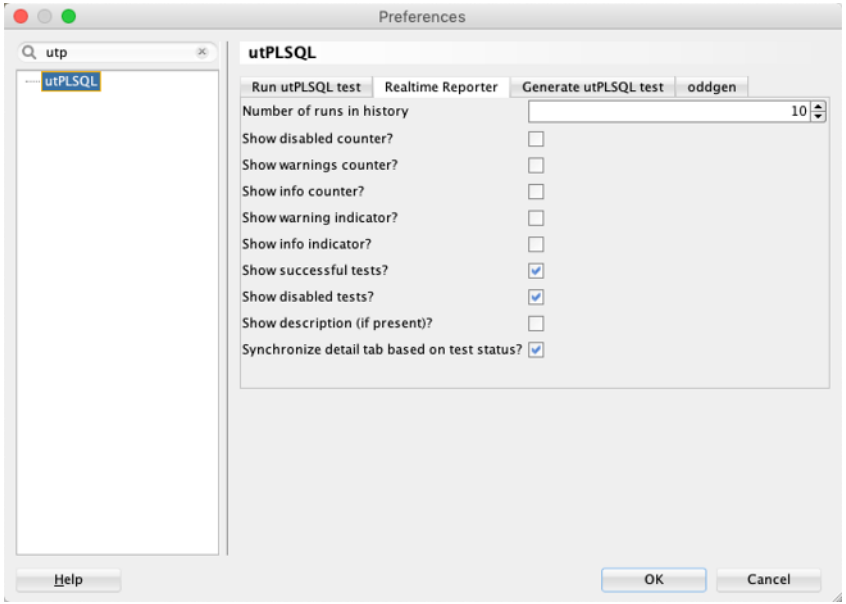




Configuration – Running utPLSQL Tests

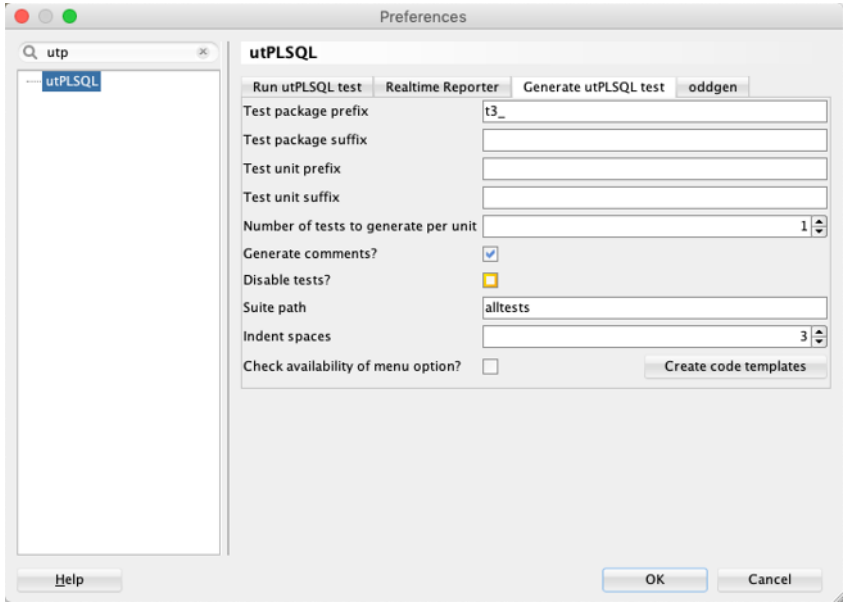
trivadis

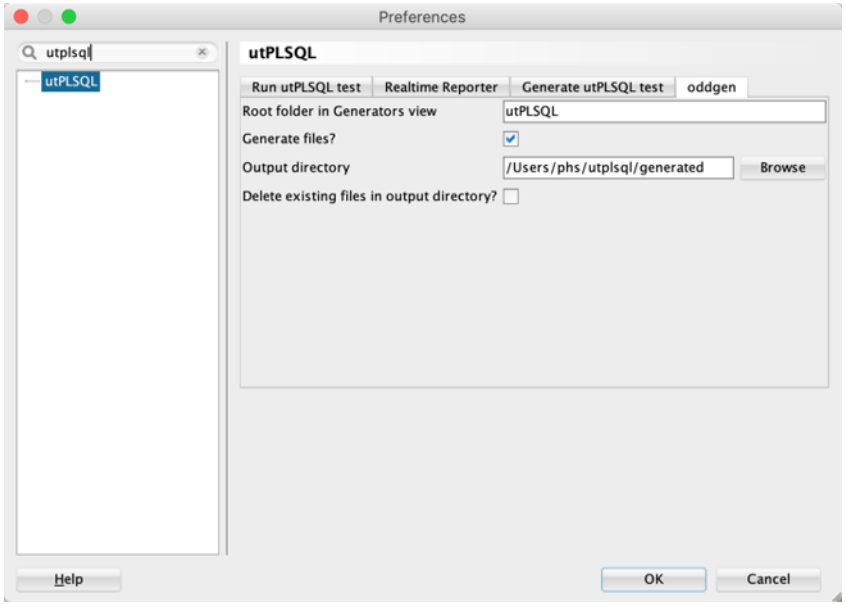




Configuration – Generating utPLSQL Tests

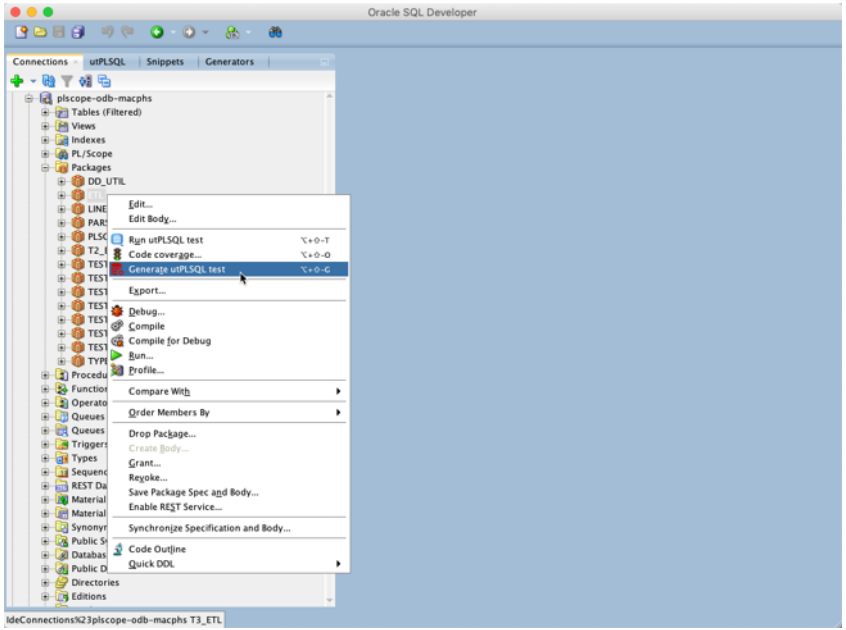
trivadis

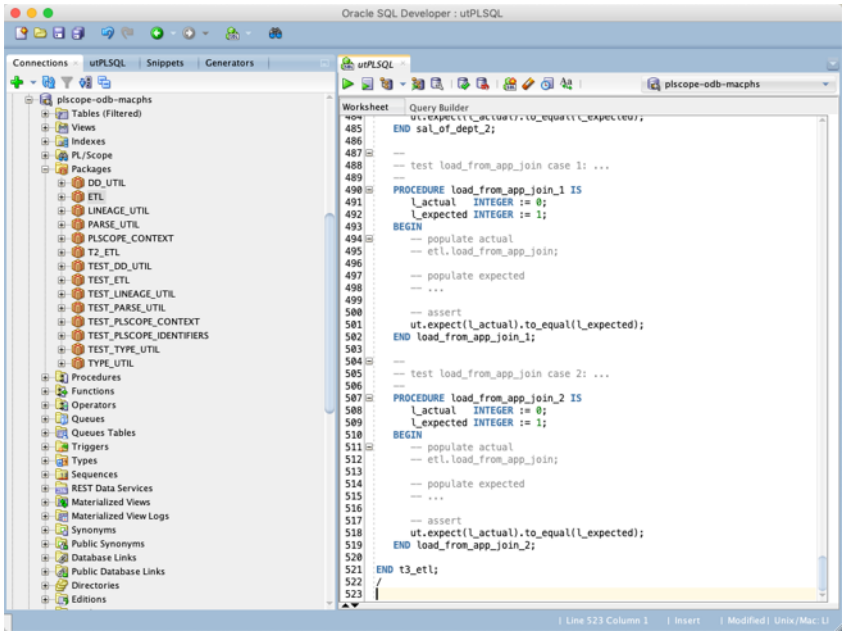




Test Last – Create Test from Existing Code

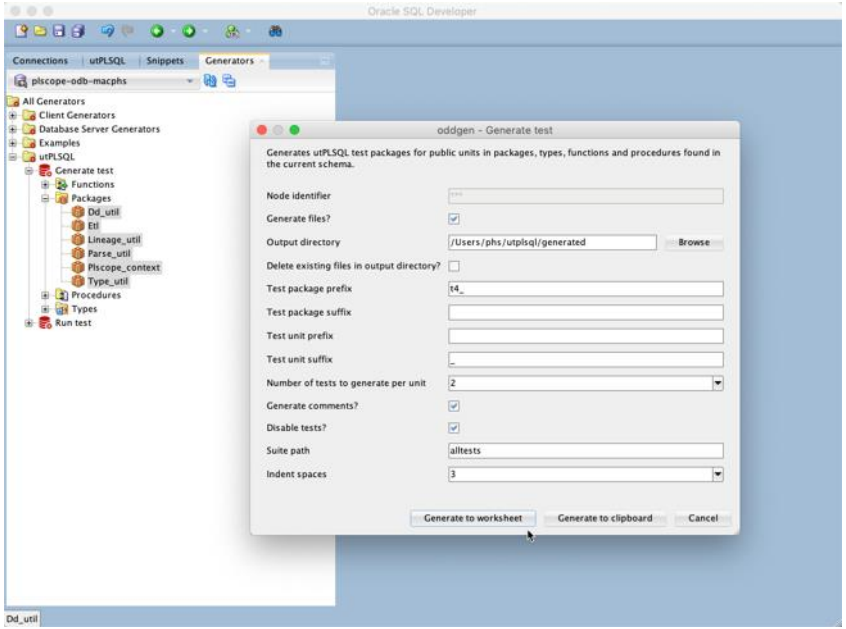
trivadis

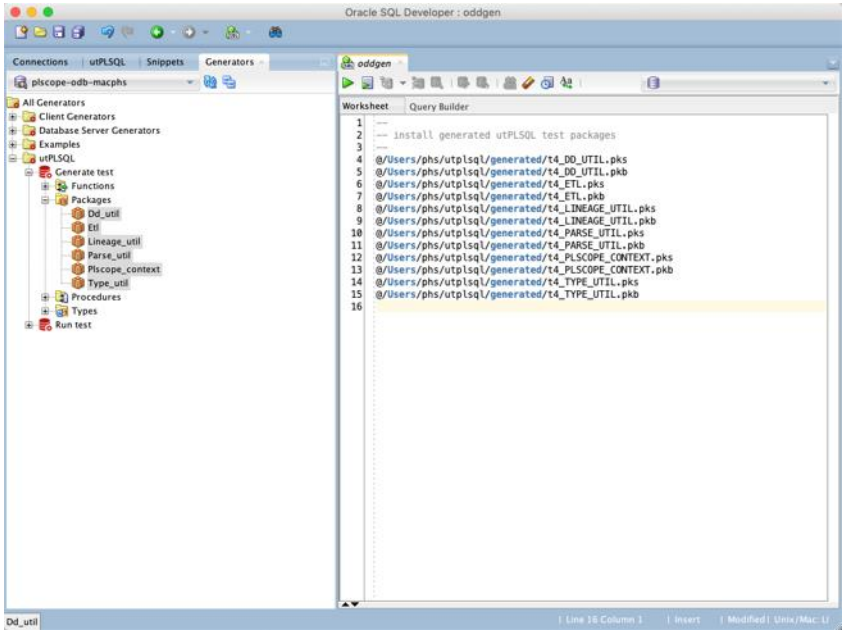




Test Last – Generate Multiple Test Skeletons

trivadis





trivadis

Run Code Coverage Reports in SQL Developer

Code Coverage – Defintion

trivadis

A measure used to describe the
degree to which the source code of a program is executed
when a particular test suite runs.

Source: https://en.wikipedia.org/wiki/Code_coverage

Line Coverage

trivadis

```
CREATE OR REPLACE FUNCTION f(a IN INTEGER) RETURN INTEGER IS
BEGIN
    IF a IS NULL THEN
        RETURN 0;
    ELSE
        RETURN a*a;
    END IF;
END f;
/
```

Two test cases for 100% coverage

Code Block Coverage (12.2 and higher)

trivadis

```
CREATE OR REPLACE FUNCTION f(a IN INTEGER) RETURN INTEGER IS
BEGIN
   IF a IS NULL THEN RETURN 0; ELSE RETURN a*a; END IF;
END f;
/
```

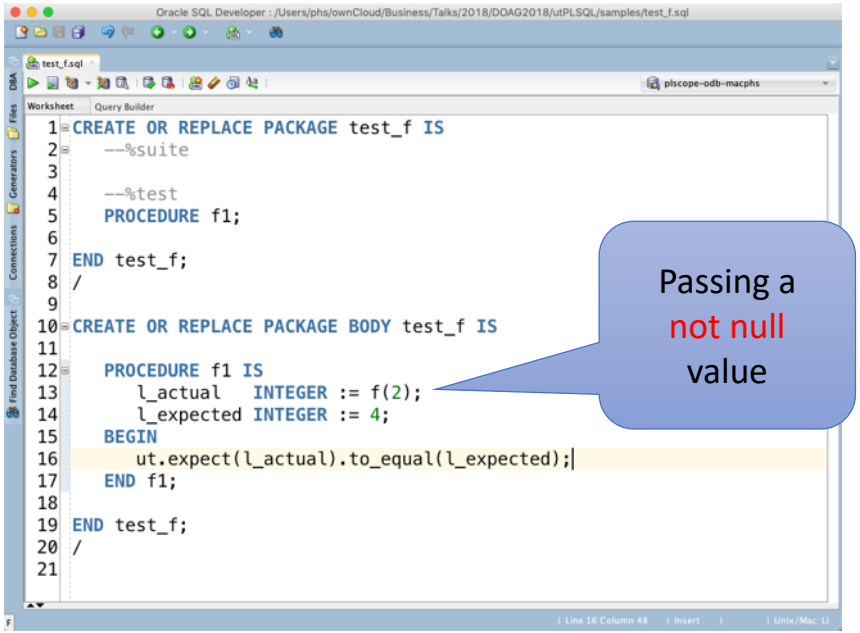
Two test cases for 100% coverage

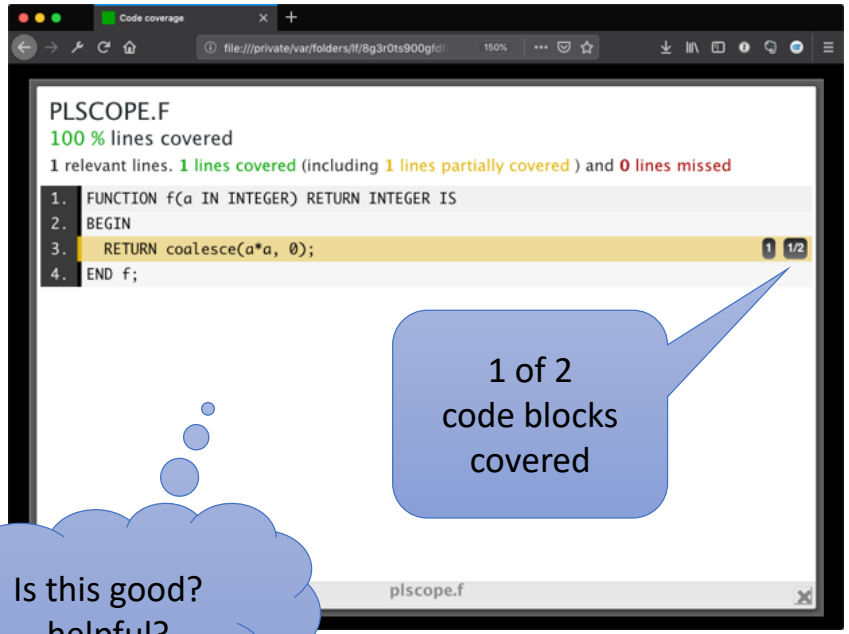
```
CREATE OR REPLACE FUNCTION f(a IN INTEGER) RETURN INTEGER IS
BEGIN
   RETURN coalesce(a*a, 0);
END f;
/
```

One test case for 100% coverage when passing NULL

utPLSQL – Line & Code Block Coverage

trivadis

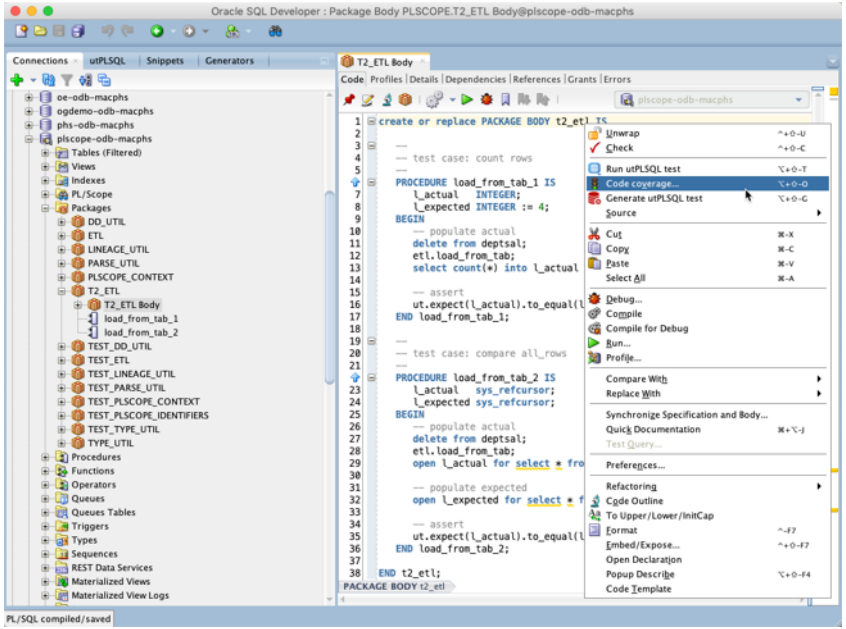


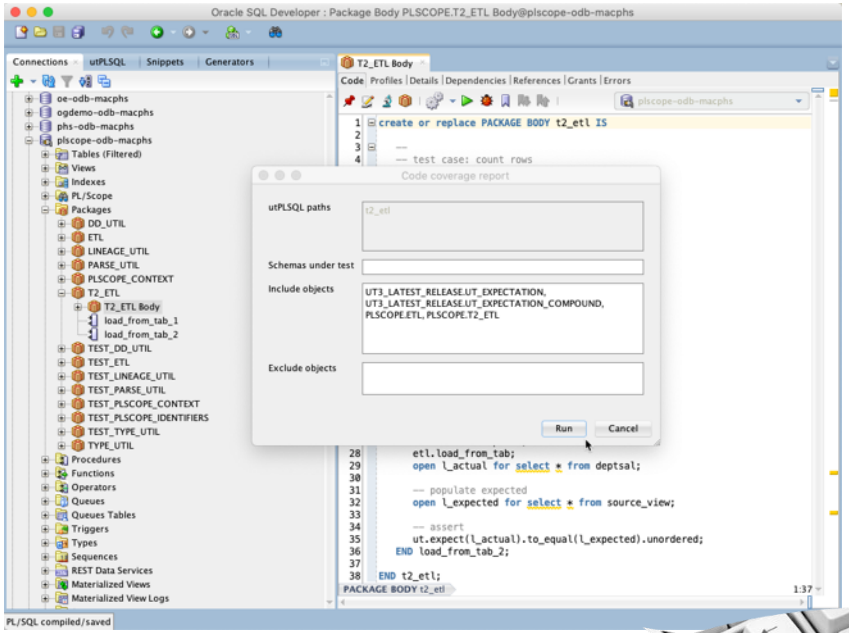


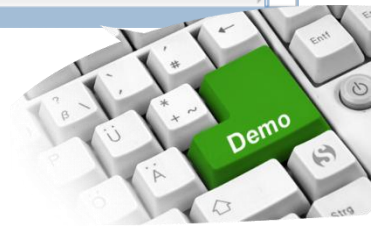
Is this good? helpful?

Run Code Coverage Report

trivadis

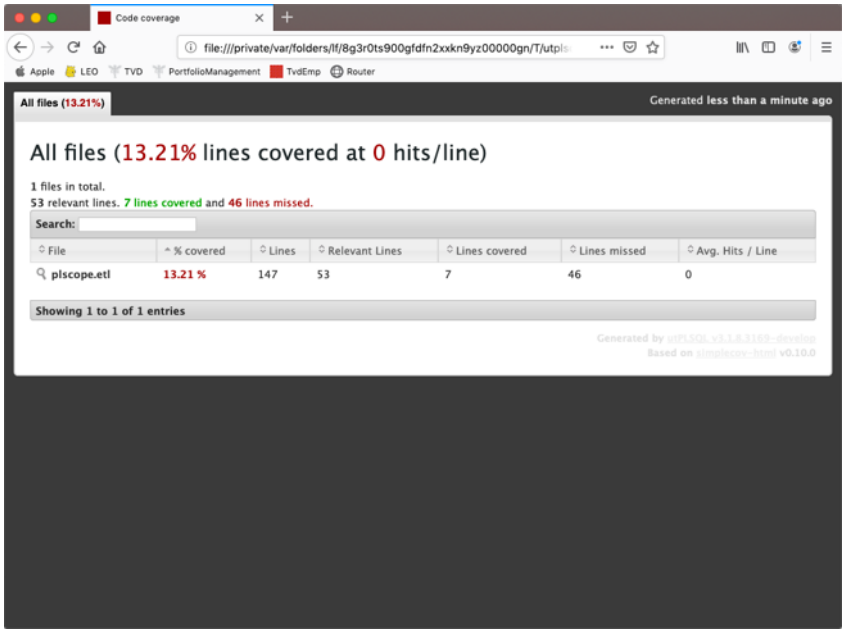


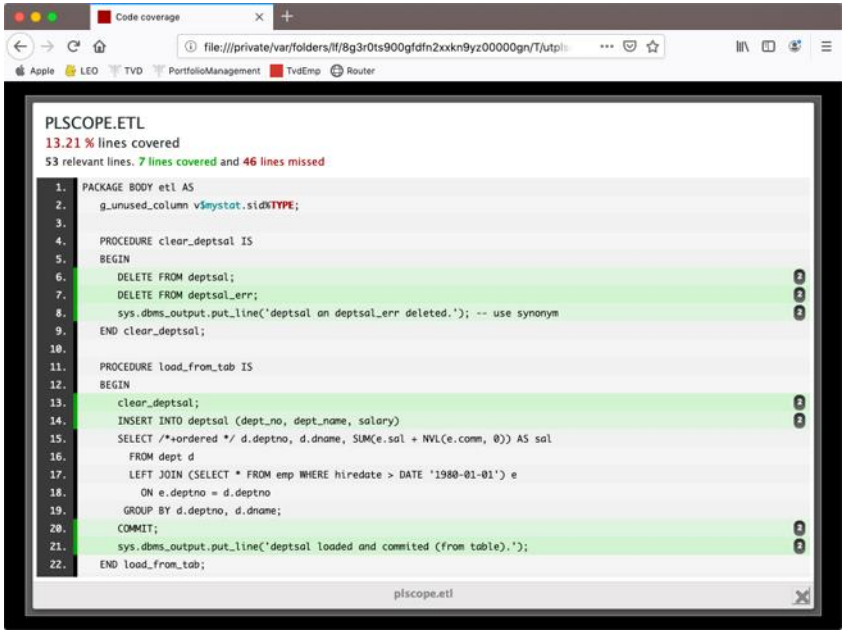




Code Coverage Report

trivadis





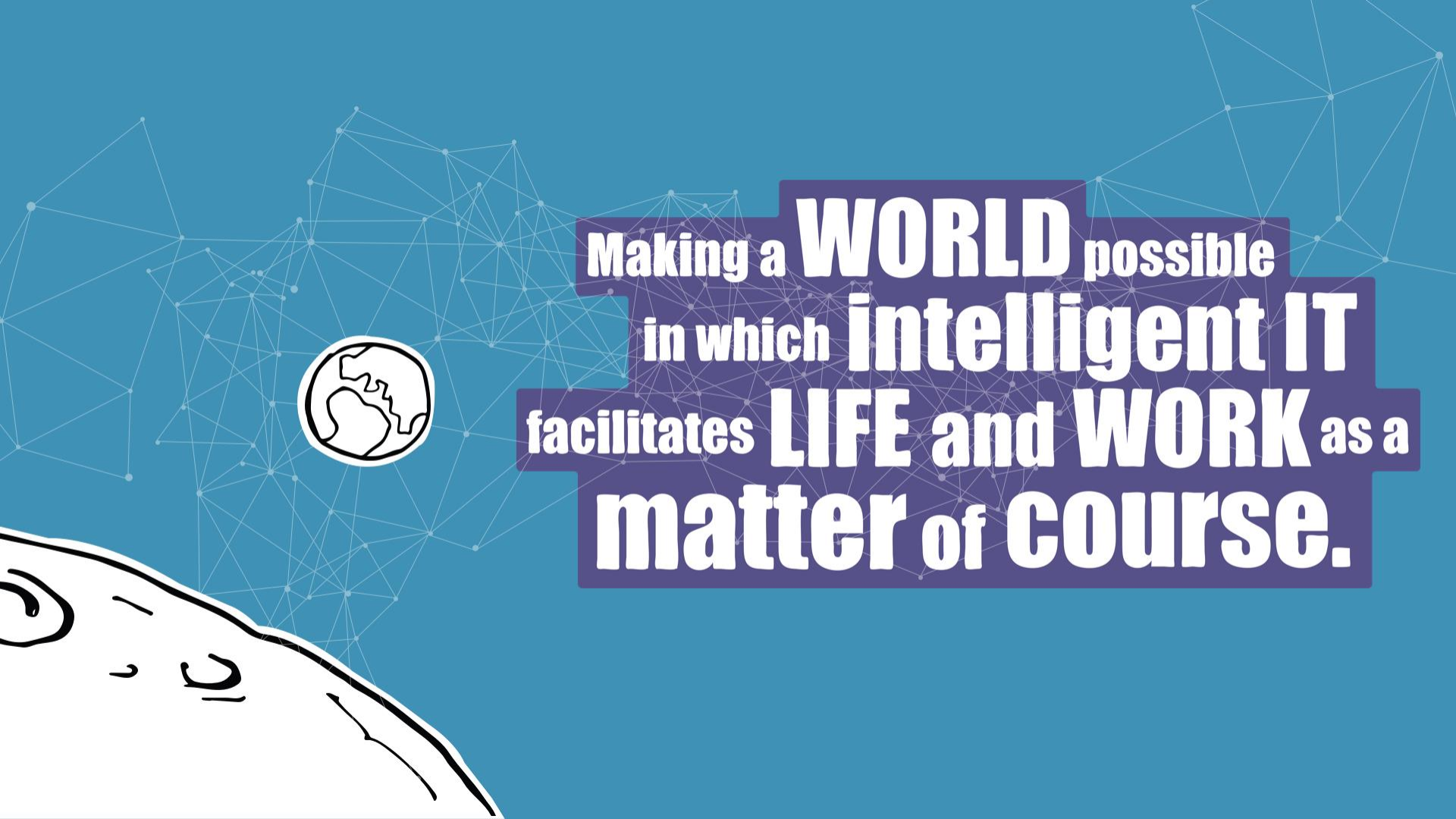
trivadis

Core Messages

The First Step Is the Hardest

trivadis

- **Set up a test-friendly environment**
  - Install utPLSQL core testing framework
  - Install SQL Developer for utPLSQL
- **Start with tests**
  - to reproduce bugs
  - for new requirements
- **utPLSQL will change how you code**
  - Write smaller units
  - Isolate code that is difficult to test


Making a WORLD possible
in which intelligent IT
facilitates LIFE and WORK as a
matter of course.