Fighting Bad PL/SQL & SQL with VS Code

Philipp Salvisberg
13th May 2025

GRISSELBAV
simple & accurate IT solutions.

Welcome

Philipp Salvisberg
Founder, Owner and CEO of Grisselbav GmbH

- Database-centric Development
- Model-driven Software Development
- Open-Source Development

philipp.salvisberg@grisselbav.com
https://www.salvis.com/blog/

Where Is My Code

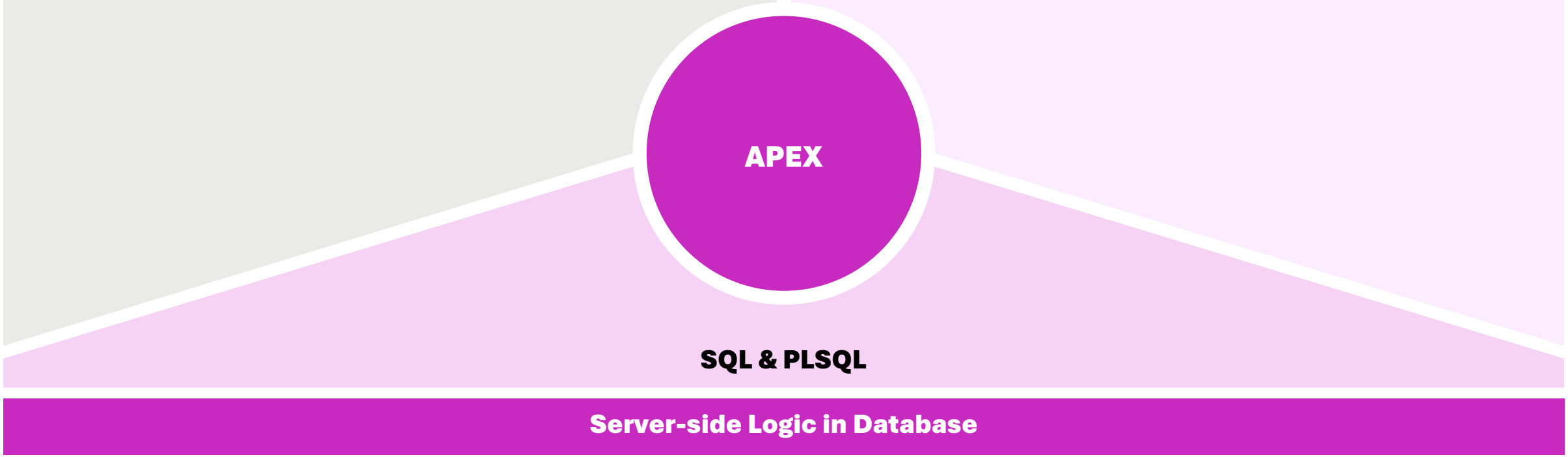
Application

User Interface

- Forms
- Reports

Client-side Logic

- Processes
- Validations



SQL

Original Report in APEX

```
select d.deptno, d.dname, …
  from dept d
  join emp e
    on e.deptno = d.deptno
  join salgrade s …
  join …
  join …
 where d.deptno = :p10_deptno
   and …
```

Refactoring

Relational View in DB

```
create view dept_emp_v as
select d.deptno, d.dname, …
  from dept d
  join emp e …
```

Simplified Report in APEX

```
select deptno, dname, …
  from dept_emp_v
 where deptno = :p10_deptno
   and …
```

PL/SQL

Original Process in APEX

```
declare
   l_comm emphist.comm%type;
   …
begin
   select h.comm into l_comm
     from …
    where e.id = :p20_id;
   if l_comm > 4000 then
      …
   else
      case …
      end case;
   end if;
   :p20_comm := l_comm;
end;
```

Refactoring

PL/SQL Package in DB

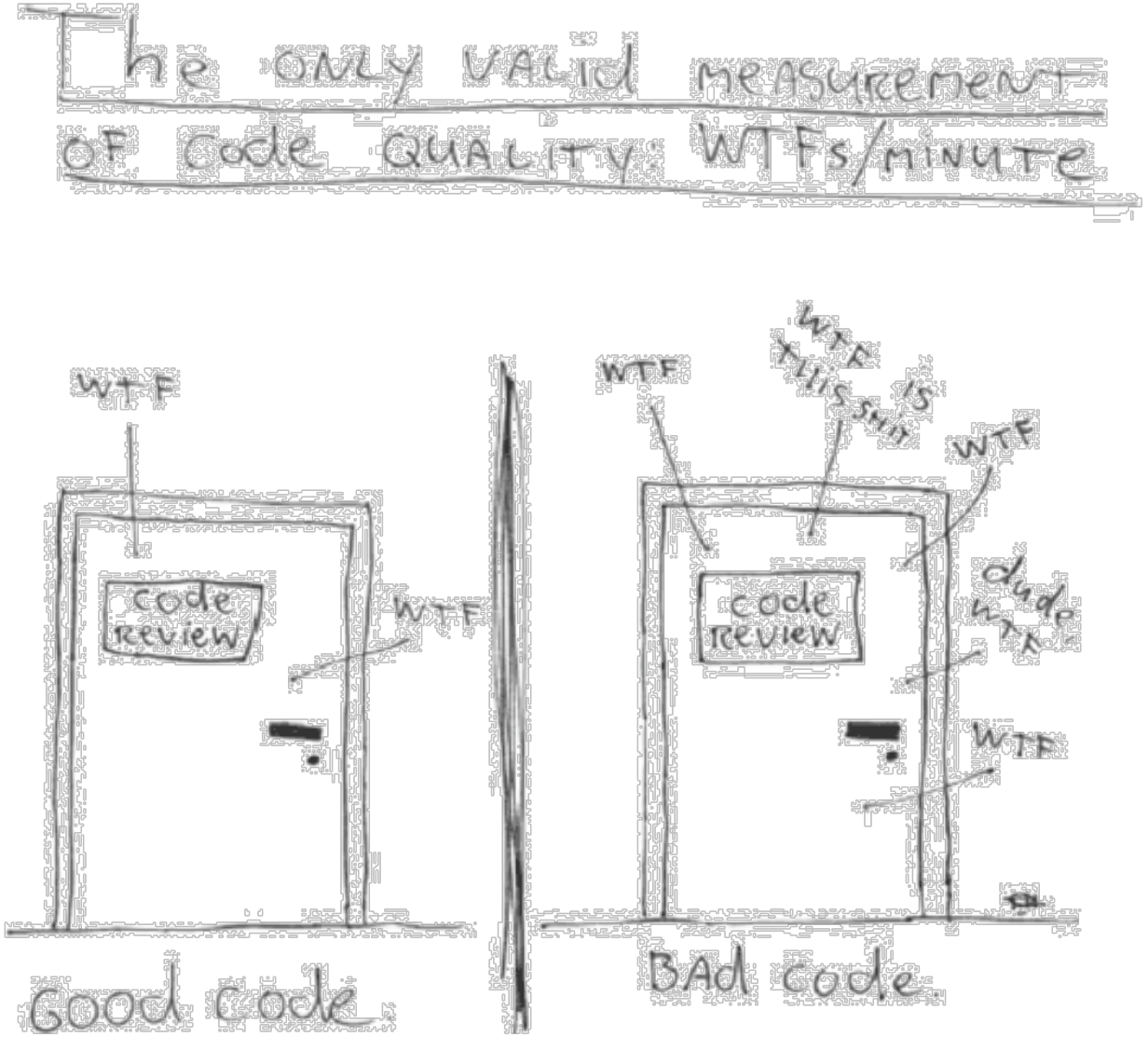
```
create package emp_mgmt is
   function comm(in_id in integer)
   return number is …
end;
create package body emp_mgmt is
   function comm(in_id in integer)
   return number is …
   begin
      …
   end; …
end;
```

Simplified Process in APEX

```
:p20_comm := emp_mgmt.comm(:p20_id);
```

Software Quality

Code Reviews



Source: https://www.osnews.com/story/19266/wtfsm/

Tips

Source: https://www.youtube.com/watch?v=IhE4pTprQ4E

PL/SQL & SQL Coding Guidelines

The Oracle Database Developer community is made stronger by resources freely shared by experts around the world, such as the Trivadis Coding Guidelines. If you have not yet adopted standards for writing SQL and PL/SQ
in your applications, this is a great place to start.

Every line you don't write, is a line you don't have to maintain

Steven Feuerstein

Steven Feuerstein
Senior Advisor
Insum Solutions

Code is more often read than written

Coding Guidelines are a crucial part o
fact, that code is more often read th written
efforts to ease the work of th reader, which is not necessarily the author.

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

Roger Troller
Senior Consultant
finnova AG Bankware

Source: https://trivadis.github.io/plsql-and-sql-coding-guidelines/v4.4/

Naming Conventions

G-9102 FREE

Warning

Always follow naming conventions for local variables.

✕ Non-Compliant Example

```
1  declare
2     some_name integer;
3  begin
4     null;
5  end;
6  /
```

✓ Compliant Solution - ★★★★★

```
1  declare
2     l_some_name integer;
3  begin
4     null;
5  end;
6  /
```

Parameters

Use parameters to customize the rule to your needs.

Parameter	Description	Default Value
ObjectTypes	Comma-separated list of unqualified database object types. If empty, the installed object types are used.	,
CollectionPattern	Regular expression pattern for PL/SQL local variables of type array/table.	(?i)^t_[a-z0-9$#_]+$
ObjectPattern	Regular expression pattern for PL/SQL local variable of SQL object type.	(?i)^o_[a-z0-9$#_]+$
CollectionTypes	Comma-separated list of unqualified database collection types. If empty, the installed collection types are used.	,
LocalVariablePattern	Regular expression pattern for PL/SQL common local variables.	(?i)^l_[a-z0-9$#_]+$
RecordPattern	Regular expression pattern for PL/SQL local variables of type record.	(?i)^r_[a-z0-9$#_]+$
CursorPattern	Regular expression pattern for PL/SQL local variables of type cursor.	(?i)^c_[a-z0-9$#_]+$

Source: https://dblinter-rules.united-codes.com/rules/G-9102/

Coding Style

```
begin for rec in (select r.country_region as region, p.prod_category,
sum (s.amount_sold) as amount_sold from sales s join products p on p.
prod_id = s .prod_id join customers cust on cust .cust_id = s.cust_id
join times t on t.time_id= s.time_id join countries r on r.country_id
= cust .country_id where calendar_year=2022 group by r.country_region
, p . prod_category order by r . country_region , p . prod_category )
loop if rec.region = 'Europe'  then if rec . prod_category = 'Tennis'
then /* print only one line for demo purposes */  sys . dbms_output .
put_line('Amount: '||rec.amount_sold );end if; end if; end loop; end;
/
```

```
begin
   for rec in (
      select r.country_region as region,
             p.prod_category,
             sum(s.amount_sold) as amount_sold
        from sales s
        join products p
          on p.prod_id = s.prod_id
        join customers cust
          on cust.cust_id = s.cust_id
        join times t
          on t.time_id = s.time_id
        join countries r
          on r.country_id = cust.country_id
       where calendar_year = 2022
       group by r.country_region,
             p.prod_category
       order by r.country_region,
             p.prod_category
   ) loop
      if rec.region = 'Europe' then
         if rec.prod_category = 'Tennis' then /* print only one line for demo purposes */
            sys.dbms_output.put_line('Amount: ' || rec.amount_sold);
         end if;
      end if;
   end loop;
end;
/
```

Rules

G-4230 FREE

Warning

Always use a COALESCE instead of a NVL command, if parameter 2 of the NVL function is a function call or a SELECT statement.

Performance | Control

Rule Details

Severity Level	Quality Area	Fixing times	Database Support
Critical	Reliability	1 mins Quick Fix | Local-easy	OracleDB (v9.0.1 +)

Reason

The `nvl` function always evaluates both parameters before deciding which one to use. This can be harmful if parameter 2 is either a function call or a select statement, as it will be executed regardless of whether parameter 1 contains a `null` value or not.

The `coalesce` function does not have this drawback.

Example

Non-Compliant Example

```
1 select nvl(dummy,my_package.expensive_null(value_in => dummy))
2   from dual;
```

Compliant Solution - ★★★★★

```
1 select coalesce(dummy,my_package.expensive_null(value_in => dummy))
2   from dual;
```

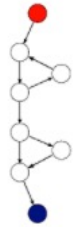
Source: https://dblinter-rules.united-codes.com/rules/G-4230/

Complexity Analysis

$$M = E - N + 2P$$

where

- M = cyclomatic complexity
- E = the number of edges of the graph
- N = the number of nodes of the graph
- P = the number of connected components.



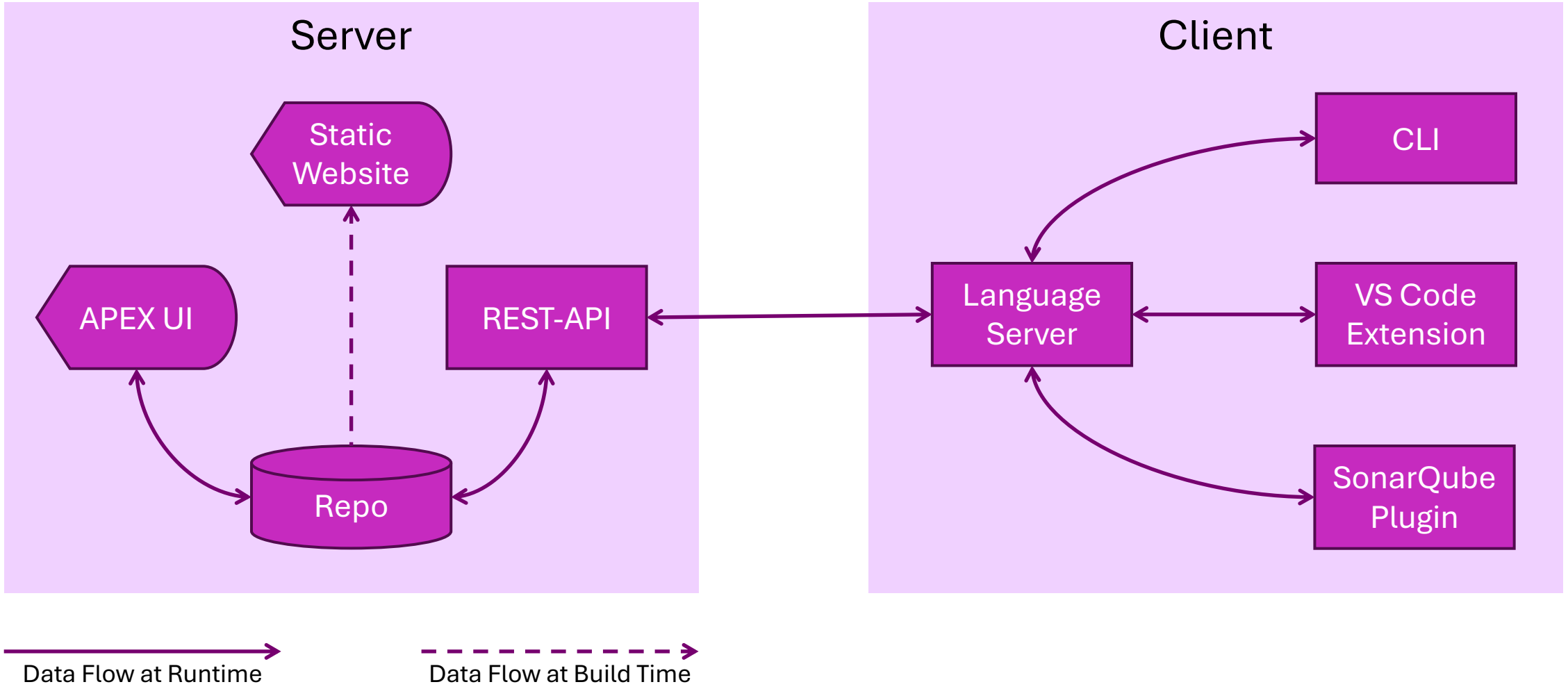
Take, for example, a control flow graph of a simple program. The program begins executing at the red node, then enters a loop (group of three nodes immediately below the red node). On exiting the loop, there is a conditional statement (group below the loop), and finally the program exits at the blue node. For this graph, $E = 9$, $N = 8$ and $P = 1$, so the cyclomatic complexity of the program is 3.

```
1  begin
2     for i in 1..3
3     loop
4        dbms_output.put_line('in loop');
5     end loop;
6     --
7     if 1 = 1
8     then
9        dbms_output.put_line('yes');
10    end if;
11    --
12    dbms_output.put_line('end');
13 end;
14 /
```

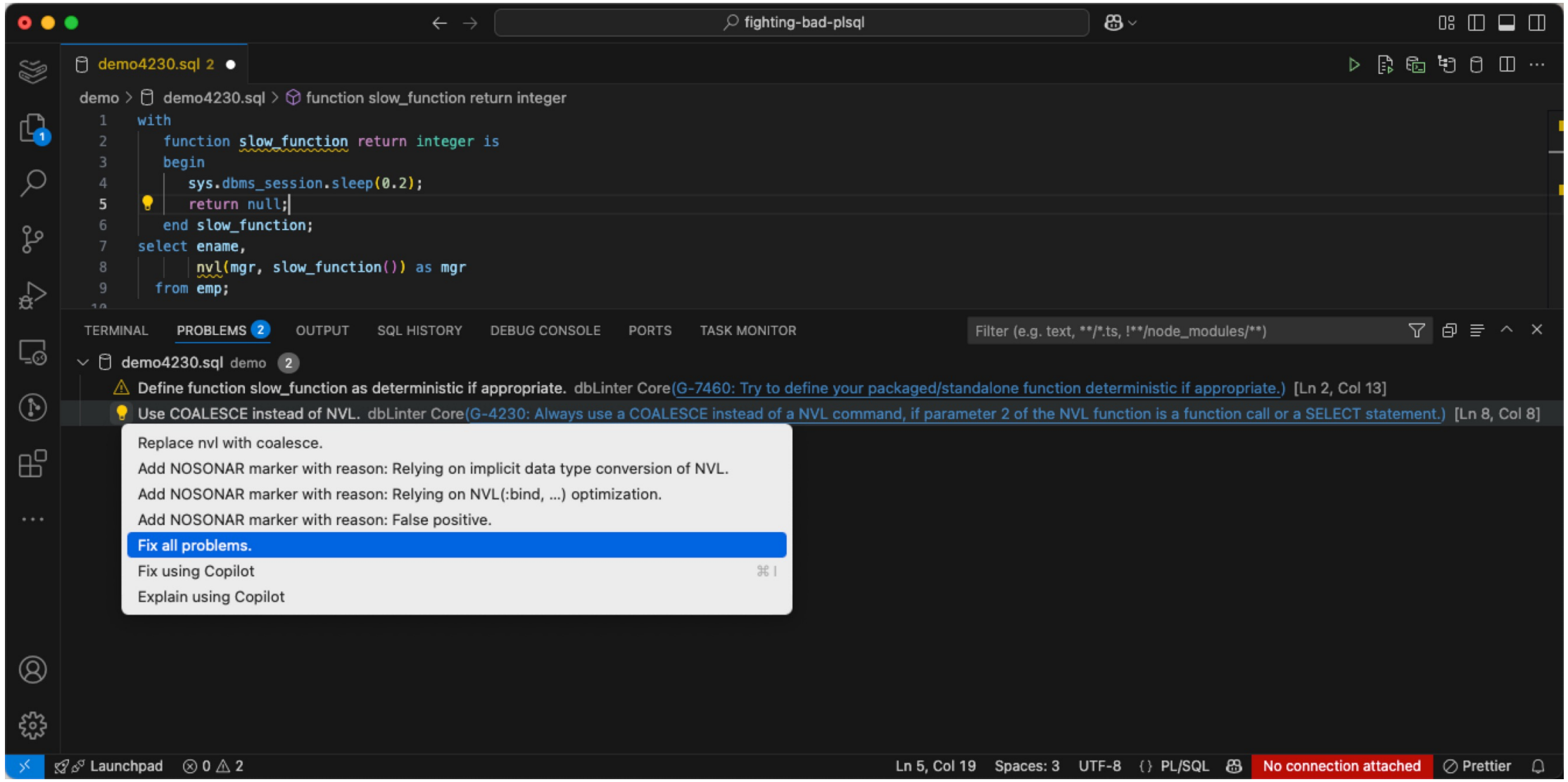

Source: https://trivadis.github.io/plsql-and-sql-coding-guidelines/main/5-complexity-analysis/complexity-analysis/

dbLinter

dbLinter Architecture

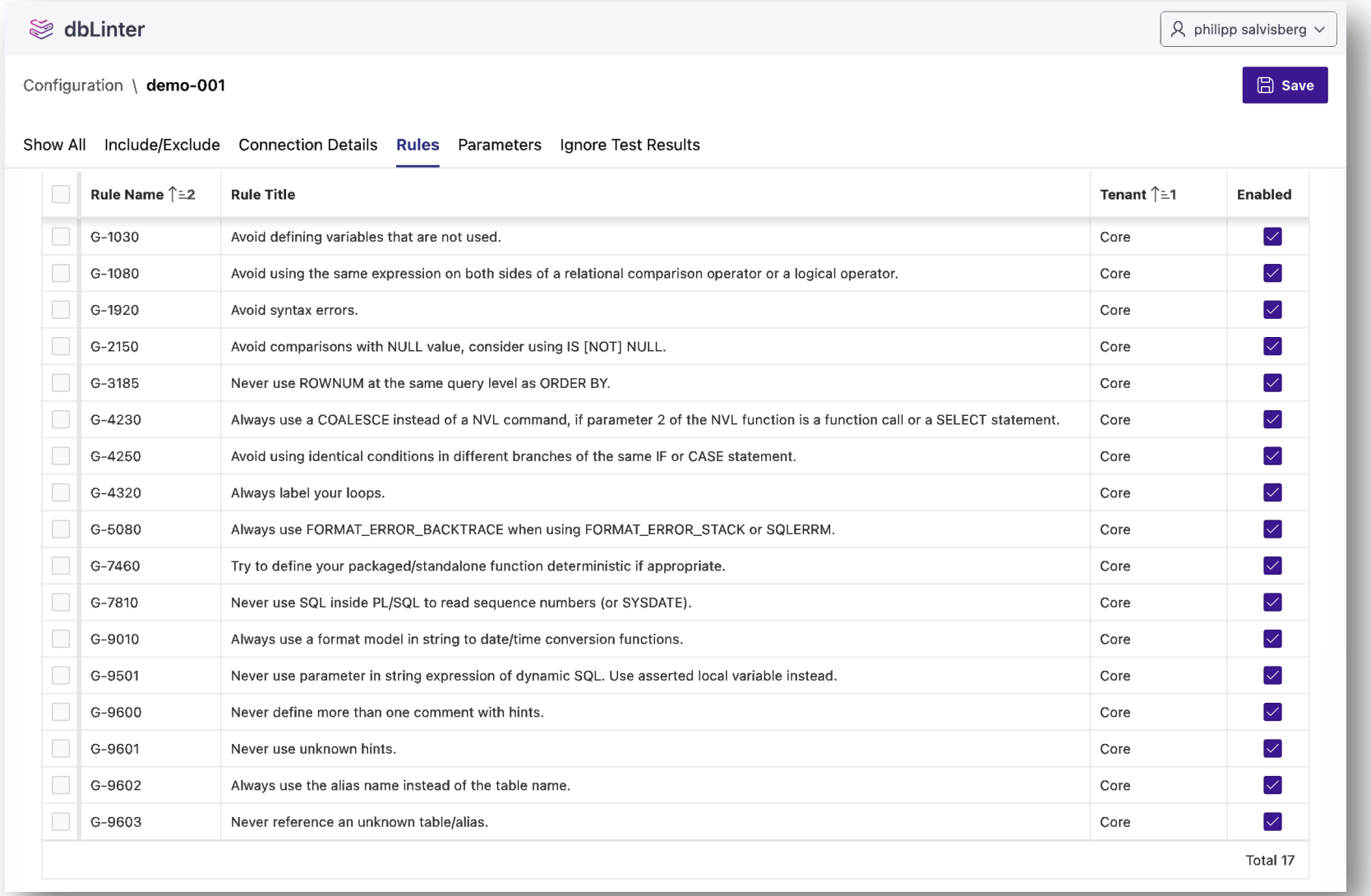


dbLinter for VS Code – Demo



Download: https://marketplace.visualstudio.com/items?itemName=Grisselbav.dblinter

Demonstrated Rules



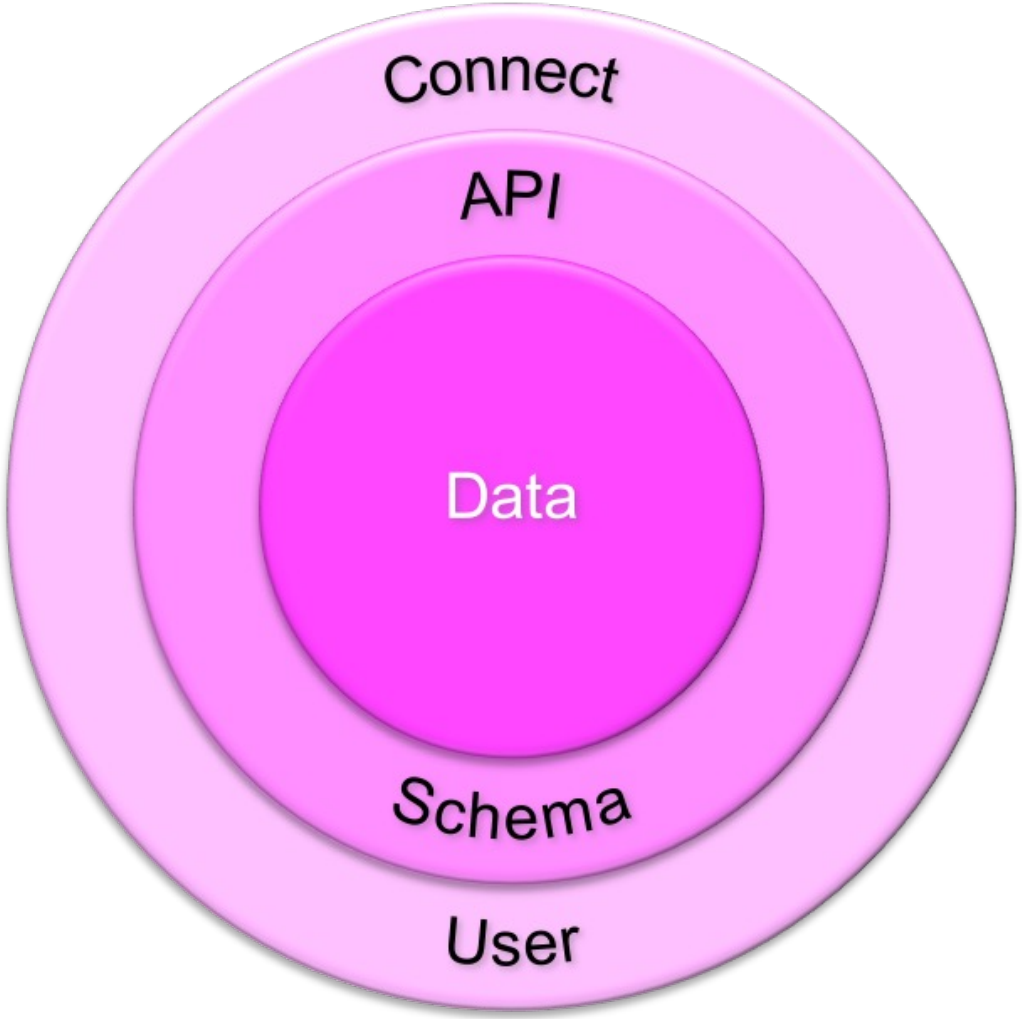
Rule Name	Rule Title	Tenant	Enabled
G-1030	Avoid defining variables that are not used.	Core	✓
G-1080	Avoid using the same expression on both sides of a relational comparison operator or a logical operator.	Core	✓
G-1920	Avoid syntax errors.	Core	✓
G-2150	Avoid comparisons with NULL value, consider using IS [NOT] NULL.	Core	✓
G-3185	Never use ROWNUM at the same query level as ORDER BY.	Core	✓
G-4230	Always use a COALESCE instead of a NVL command, if parameter 2 of the NVL function is a function call or a SELECT statement.	Core	✓
G-4250	Avoid using identical conditions in different branches of the same IF or CASE statement.	Core	✓
G-4320	Always label your loops.	Core	✓
G-5080	Always use FORMAT_ERROR_BACKTRACE when using FORMAT_ERROR_STACK or SQLERRM.	Core	✓
G-7460	Try to define your packaged/standalone function deterministic if appropriate.	Core	✓
G-7810	Never use SQL inside PL/SQL to read sequence numbers (or SYSDATE).	Core	✓
G-9010	Always use a format model in string to date/time conversion functions.	Core	✓
G-9501	Never use parameter in string expression of dynamic SQL. Use asserted local variable instead.	Core	✓
G-9600	Never define more than one comment with hints.	Core	✓
G-9601	Never use unknown hints.	Core	✓
G-9602	Always use the alias name instead of the table name.	Core	✓
G-9603	Never reference an unknown table/alias.	Core	✓

Total 17

see also: https://dblinter-rules.united-codes.com/severity-levels/blocker/

Where Is My Code Again?

SmartDB & PinkDB – Two of a Kind



Data is more often read than written

Consistent data simplifies the work of consumers

see PinkDB rules: https://dblinter-rules.united-codes.com/tags/pinkdb/

Thank you!

GRISSELBAV

simple & accurate IT solutions.