sonar RULES Products

- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C#
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Kubernetes
- Objective C
- PHP
- PL/I
- PL/SQL
- Python
- RPG
- Ruby
- Scala
- Swift
- Terraform
- Text
- TypeScript
- **T-SQL**
- VB.NET
- VB6
- XML

T-SQL static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your T-SQL code

All rules 80 | Vulnerability 1 | Bug 16 | Security Hotspot 4 | Code Smell 59

Tags | Search by name...

- Columns to be read with a "SELECT" statement should be clearly defined — Code Smell
- "CASE" expressions should not have too many "WHEN" clauses — Code Smell
- Sections of code should not be commented out — Code Smell
- Unused procedure and function parameters should be removed — Code Smell
- Track uses of "FIXME" tags — Code Smell
- Redundant pairs of parentheses should be removed — Code Smell
- Functions and procedures should not have too many parameters — Code Smell
- Collapsible "if" statements should be merged — Code Smell
- Unused labels should be removed — Code Smell
- Using hardcoded IP addresses is security-sensitive — Security Hotspot
- Column references should not have more than two-parts — Code Smell

Columns to be read with a "SELECT" statement should be clearly defined

Analyze your code

Code Smell   Major   performance sql

`SELECT *` should be avoided because it releases control of the returned columns and could therefore lead to errors and potentially to performance issues.

Noncompliant Code Example

```
SELECT *     -- Noncompliant
       FROM persons
       WHERE city = 'NEW YORK'
```

Compliant Solution

```
SELECT firstname, lastname
       FROM persons
       WHERE city = 'NEW YORK'
```

Exceptions

The following cases are ignored by this rule:

- `SELECT` from temporary tables: `SELECT * FROM #temp1`
- `SELECT` using common table expressions: `WITH A AS (SELECT C1 FROM T1) SELECT * FROM A;`
- Inside another `SELECT`: `SELECT C1 FROM T1 WHERE C2 IN (SELECT * FROM T2)`
- Inside `INSERT`: `INSERT INTO T1 SELECT * FROM T2`
- Inside `CREATE TABLE`: `CREATE TABLE T1 WITH (C1 = C2) AS SELECT * FROM T2`
- `SELECT` from rowset providers: `SELECT * FROM OPENXML (@idoc, '/ROOT/Customer',1)`
- `SELECT INTO`: `SELECT * INTO NEW_TABLE FROM T1`
- `SELECT` from variable table: `SELECT * FROM @table1`
- `SELECT` from derived table: `SELECT A.* FROM (SELECT X FROM T1) A INNER JOIN B ON A.X = B.X`

Available In:

sonarlint | sonarcloud | sonarqube Developer Edition