

# CodeQL CLI commands manual

Reference information for the commands available in the most recent release of CodeQL CLI.

GitHub CodeQL is licensed on a per-user basis upon installation. You can use CodeQL only for certain tasks under the license restrictions. For more information, see "[About the CodeQL CLI](#)." If you have a GitHub Advanced Security license, you can use CodeQL for automated analysis, continuous integration, and continuous delivery. For more information, see "[About GitHub Advanced Security](#)."

This content describes the most recent release of the CodeQL CLI. For more information about this release, see <https://github.com/github/codeql-cli-binaries/releases>.

To see details of the options available for this command in an earlier release, run the command with the `--help` option in your terminal.

## bqrs decode

Convert result data from BQRS into other forms.

## bqrs diff

Compute the difference between two result sets.

## bqrs hash

[Plumbing] Compute a stable hash of a BQRS file.

## bqrs info

Display metadata for a BQRS file.

## bqrs interpret

[Plumbing] Interpret data in a single BQRS.

## database add-diagnostic

[Experimental] Add a piece of diagnostic information to a database.

## database analyze

Analyze a database, producing meaningful results in the context of the source code.

## database bundle

Create a relocatable archive of a CodeQL database.

---

### **database cleanup**

Compact a CodeQL database on disk.

---

### **database create**

Create a CodeQL database for a source tree that can be analyzed using one of the CodeQL products.

---

### **database export-diagnostics**

[Experimental] Export diagnostic information from a database for a failed analysis.

---

### **database finalize**

[Plumbing] Final steps in database creation.

---

### **database import**

[Advanced] [Plumbing] Import unfinalized database(s) into another unfinalized database.

---

### **database index-files**

[Plumbing] Index standalone files with a given CodeQL extractor.

---

### **database init**

[Plumbing] Create an empty CodeQL database.

---

### **database interpret-results**

[Plumbing] Interpret computed query results into meaningful formats such as SARIF or CSV.

---

### **database print-baseline**

[Plumbing] Print a summary of the baseline lines of code seen.

---

### **database run-queries**

[Plumbing] Run a set of queries together.

---

### **database trace-command**

[Plumbing] Run a single command as part of a traced build.

---

### **database unbundle**

Extracts a CodeQL database archive.

---

### **database upgrade**

Upgrade a database so it is usable by the current tools.

---

### **dataset check**

[Plumbing] Check a particular dataset for internal consistency.

---

### **dataset cleanup**

[Plumbing] Clean up temporary files from a dataset.

---

### **dataset import**

[Plumbing] Import a set of TRAP files to a raw dataset.

---

### **dataset measure**

[Plumbing] Collect statistics about the relations in a particular dataset.

---

### **dataset upgrade**

[Plumbing] Upgrade a dataset so it is usable by the current tools.

---

### **diagnostic add**

[Experimental] [Plumbing] Add a piece of diagnostic information.

---

### **diagnostic export**

[Experimental] Export diagnostic information for a failed analysis.

---

### **execute cli-server**

[Deep plumbing] Server for running multiple commands while avoiding repeated JVM initialization.

---

### **execute language-server**

[Plumbing] On-line support for the QL language in IDEs.

---

### **execute queries**

[Plumbing] Run one or more queries against a dataset.

---

### **execute query-server**

[Plumbing] Support for running queries from IDEs.

---

### **execute query-server2**

[Plumbing] Support for running queries from IDEs.

---

## **execute upgrades**

[Plumbing] Run upgrade scripts on an existing raw QL dataset.

---

## **generate extensible-predicate-metadata**

[Experimental] [Deep plumbing] Report the extensible predicates found in the given pack.

---

## **generate log-summary**

[Advanced] Create a summary of a structured log file.

---

## **generate query-help**

Generate end-user query help from .qhelp files.

---

## **github upload-results**

Uploads a SARIF file to GitHub code scanning.

---

## **pack add**

[Experimental] Adds a list of QL library packs with optional version ranges as dependencies of the current package, and then installs them.

---

## **pack bundle**

[Experimental] [Plumbing] Bundle a QL library pack.

---

## **pack ci**

[Experimental] Install dependencies for this pack, verifying that the existing lock file is up to date.

---

## **pack create**

[Experimental] [Plumbing] Builds the contents of a QL package from source code.

---

## **pack download**

[Experimental] Download the set of qlpacks referenced by the query spec of the command line from the registry. Packs can be provided by name or implicitly inside of a query suite (.qls) file.

---

## **pack init**

[Experimental] Initializes a qlpack in the specified directory.

---

## **pack install**

[Experimental] Install dependencies for this pack.

---

## **pack ls**

[Experimental] [Deep plumbing] List the CodeQL packages rooted at this directory. This directory must contain a qlpack.yml or .codeqlmanifest.json file.

---

## **pack packlist**

[Experimental] [Plumbing] Compute the set of files to be included in a QL query pack or library pack.

---

## **pack publish**

[Experimental] Publishes a QL library pack to a package registry.

---

## **pack resolve-dependencies**

[Experimental] [Plumbing] Compute the set of required dependencies for this QL pack.

---

## **pack upgrade**

[Experimental] Update the dependencies for this pack to the latest available versions.

---

## **query compile**

Compile or check QL code.

---

## **query decompile**

[Plumbing] Read an intermediate representation of a compiled query from a .qlo file.

---

## **query format**

Autoformat QL source code.

---

## **query run**

Run a single query.

---

## **resolve database**

[Deep plumbing] Report metadata about the database.

---

## **resolve extensions**

[Experimental] [Deep plumbing] Determine accessible extensions. This includes machine learning models and data extensions.

---

## **resolve extensions-by-pack**

[Experimental] [Deep plumbing] Determine accessible extensions for the given paths to pack roots. This includes machine learning models and data extensions.

---

## resolve extractor

[Deep plumbing] Determine the extractor pack to use for a given language.

---

## resolve files

[Deep plumbing] Expand a set of file inclusion/exclusion globs.

---

## resolve languages

List installed CodeQL extractor packs.

---

## resolve library-path

[Deep plumbing] Determine QL library path and dbscheme for a query.

---

## resolve metadata

[Deep plumbing] Resolve and return the key-value metadata pairs from a query source file.

---

## resolve ml-models

[Deprecated] [Experimental] [Deep plumbing] Determine accessible machine learning models.

---

## resolve qlpacks

Create a list of installed QL packs and their locations.

---

## resolve qlref

[Deep plumbing] Dereferences a .qlref file to return a .ql one.

---

## resolve queries

[Deep plumbing] Expand query directories and suite specifications.

---

## resolve ram

[Deep plumbing] Prepare RAM options.

---

## resolve tests

[Deep plumbing] Find QL unit tests in given directories.

---

## resolve upgrades

[Deep plumbing] Determine upgrades to run for a raw dataset.

---

## test accept

Accept results of failing unit tests.

---

**test extract**

[Plumbing] Build a dataset for a test directory.

---

**test run**

Run unit tests for QL queries.

---

**version**

Show the version of the CodeQL toolchain.

---

**Legal**

© 2023 GitHub, Inc. [Terms](#) [Privacy](#) [Status](#) [Pricing](#) [Expert services](#) [Blog](#)