# Report Database

Black Duck 2024.7.1

04-09-2024

# Contents

# Preface

## Black Duck documentation

The documentation for Black Duck consists of online help and these documents:

| Title | File | Description |
|---|---|---|
| Release Notes | release_notes.pdf | Contains information about the new and improved features, resolved issues, and known issues in the current and previous releases. |
| Installing Black Duck using Docker Swarm | install_swarm.pdf | Contains information about installing and upgrading Black Duck using Docker Swarm. |
| Installing Black Duck using Kubernetes | install_kubernetes.pdf | Contains information about installing and upgrading Black Duck using Kubernetes. |
| Installing Black Duck using OpenShift | install_openshift.pdf | Contains information about installing and upgrading Black Duck using OpenShift. |
| Getting Started | getting_started.pdf | Provides first-time users with information on using Black Duck. |
| Scanning Best Practices | scanning_best_practices.pdf | Provides best practices for scanning. |
| Getting Started with the SDK | getting_started_sdk.pdf | Contains overview information and a sample use case. |
| Report Database | report_db.pdf | Contains information on using the report database. |
| User Guide | user_guide.pdf | Contains information on using Black Duck's UI. |

The installation methods for installing Black Duck software in a Kubernetes or OpenShift environment are Helm. Click the following links to view the documentation.

- Helm is a package manager for Kubernetes that you can use to install Black Duck. Black Duck supports Helm3 and the minimum version of Kubernetes is 1.13.

Black Duck integration documentation is available on:

- https://sig-product-docs.synopsys.com/bundle/integrations-detect/page/integrations/integrations.html
- https://sig-product-docs.synopsys.com/category/cicd_integrations

## Customer support

If you have any problems with the software or the documentation, please contact Black Duck Customer Support.

You can contact Black Duck Support in several ways:

- Online: https://www.synopsys.com/software-integrity/support.html
- Phone: See the Contact Us section at the bottom of our support page to find your local phone number.

To open a support case, please log in to the Black Duck Software Integrity Community site at https://community.synopsys.com/s/contactsupport.

Another convenient resource available at all times is the online customer portal.

# Black Duck Software Integrity Community

The Black Duck Software Integrity Community is our primary online resource for customer support, solutions, and information. The Community allows users to quickly and easily open support cases and monitor progress, learn important product information, search a knowledgebase, and gain insights from other Software Integrity Group (SIG) customers. The many features included in the Community center around the following collaborative actions:

- Connect – Open support cases and monitor their progress, as well as, monitor issues that require Engineering or Product Management assistance
- Learn – Insights and best practices from other SIG product users to allow you to learn valuable lessons from a diverse group of industry leading companies. In addition, the Customer Hub puts all the latest product news and updates from Black Duck at your fingertips, helping you to better utilize our products and services to maximize the value of open source within your organization.
- Solve – Quickly and easily get the answers you're seeking with the access to rich content and product knowledge from SIG experts and our Knowledgebase.
- Share – Collaborate and connect with Software Integrity Group staff and other customers to crowdsource solutions and share your thoughts on product direction.

Access the Customer Success Community. If you do not have an account or have trouble accessing the system, click here to get started, or send an email to community.manager@synopsys.com.

# Training

Black Duck Software Integrity Group (SIG), Customer Education is a one-stop resource for all your Black Duck education needs. It provides you with 24x7 access to online training courses and how-to videos.

New videos and courses are added monthly.

At Black Duck Software Integrity Group (SIG), Customer Education, you can:

- Learn at your own pace.
- Review courses as often as you wish.
- Take assessments to test your skills.
- Print certificates of completion to showcase your accomplishments.

Learn more at https://community.synopsys.com/s/education or for help with Black Duck, select **Black Duck Tutorials** from the Help menu (  ) in the Black Duck UI.

# Black Duck Statement on Inclusivity and Diversity

Black Duck is committed to creating an inclusive environment where every employee, customer, and partner feels welcomed. We are reviewing and removing exclusionary language from our products and supporting customer-facing collateral. Our effort also includes internal initiatives to remove biased language from our engineering and working environment, including terms that are embedded in our software and IPs. At the same time, we are working to ensure that our web content and software applications are usable to people of varying abilities. You may still find examples of non-inclusive language in our software or documentation as our IPs implement industry-standard specifications that are currently under review to remove exclusionary language.

# Black Duck Security Commitments

As an organization dedicated to protecting and securing our customers' applications, Black Duck is equally committed to our customers' data security and privacy. This statement is meant to provide Black Duck customers and prospects with the latest information about our systems, compliance certifications, processes, and other security-related activities.

This statement is available at: Security Commitments | Black Duck

# 1. About the report database

Reporting schemas in the PostgreSQL database, `bds_hub`, provide access to Black Duck data for reporting purposes. Use any reporting tool that supports JDBC connections, such as Jasper Reports, to access the data.

With the report database, for example, you can:

- Create a report of the components in a project version.
- Create a report of the vulnerabilities in a component version.
- Query the database to obtain similar information across all of your projects, such as:
  - Selecting all projects with a particular license, phase, and/or distribution.
  - Selecting all components using a particular license.
  - Selecting all project/project versions having a particular component/component version.

**Important:** Users should not delete data from the Black Duck database (`bds_hub`) unless specifically directed to by a Black Duck Technical Support representative. Deletion of data can cause errors ranging from UI glitches to complete failure of Black Duck to start.

**Important:** Note that Black Duck Technical Support cannot recover deleted data and will only be able to suggest to restore from database backups. If no backups are available, support from Black Duck will be best-effort only.

Note the following:

- Database name: `bds_hub`
- Username: blackduck_reporter. This user only has read-only access to the reporting schema of the database.
- Exposed port: 55436

  If your Black Duck server is hosted by Black Duck Software, the exposed port is 5432.
- Password for blackduck_reporter.
  - If using the database container that is automatically installed by Black Duck, set the password before connecting to the database. For more information, see the installation guide for the orchestration tool you used to install Black Duck.
  - If using an external PostgreSQL database, use your preferred PostgreSQL administration tool to configure the password.

  Once the password is set you can connect to the report database. For example, using psql:

  ```
  psql -U blackduck_reporter -p 55436 -h localhost -W bds_hub
  ```

  **Note:** There will be a delay for any changes made in Black Duck to appear in the report database. The length of time for this delay depends on the value you specified for the BLACKDUCK_REPORTING_DELAY_MINUTES environment variable, which by default, equals 8 hours. For more information, see the installation guide for the orchestration tool you used to install Black Duck.

- The schemas are created automatically when you install or upgrade Black Duck. You will not be able to query the views until after the first run of the report database job - the ReportingDatabaseTransferJob. To determine if a view is populated, run the following command:

  ```
  SELECT ispopulated FROM pg_catalog.pg_matviews WHERE schemaname = 'reporting' AND
   matviewname = '<ViewName>'
  ```

  where `ViewName` is the name of the view that interests you.

  For example, to determine if the component policies view is populated, run the following command:

  ```
  SELECT ispopulated FROM pg_catalog.pg_matviews WHERE schemaname = 'reporting' AND
   matviewname = 'component_policies'
  ```

- Black Duck recommends that you write queries with the assumption that columns will be added in the future; select specific columns, instead of using the all-columns "*" selection method.
- While Black Duck provides the blackduck_reporter user which only has read-only access to the reporting schemas of the `bds_hub` database, you can configure additional users which have the same permissions as the blackduck_reporter.

  Run the following commands, replacing `blackduck_reporter` with the username:

  ```
  GRANT USAGE ON SCHEMA reporting TO ${blackduck_reporter};
  GRANT SELECT ON ALL TABLES IN SCHEMA reporting TO ${blackduck_reporter};
  REVOKE INSERT, UPDATE, TRUNCATE, DELETE, REFERENCES ON ALL TABLES IN SCHEMA reporting
   FROM ${blackduck_reporter};
  REVOKE ALL ON SCHEMA st FROM ${blackduck_reporter};
  ```

# 2. Report Database Schema

The following section lists the tables in the report database and associated view name.

## Component table (component)

| Column | Type | Description |
|---|---|---|
| channel_release_external_namespace | text | The origin namespace for the component. |
| component_id | UUID | Component ID. |
| component_name | text | Component name. |
| component_version_id | UUID | Component version ID. |
| component_version_name | text | Component version name. |
| created_at | timestamptz | The created at time for the component, copied from the BOM.<br>Represents the first time the component was added to the BOM. |
| id | int8 | ID. |
| ignored | boolean | Indicates whether the component is ignored:<br>• "t" indicates that the component is ignored.<br>• "f" indicates that the component is not ignored. |
| license_high_count | numeric | Number of high license risk. |
| license_low_count | numeric | Number of low license risk. |
| license_medium_count | numeric | Number of medium license risk. |
| license_ok_count | numeric | Number of no license risk. |
| operational_high_count | numeric | Number of high operational risk. |
| operational_medium_count | numeric | Number of medium operation risk. |
| operational_low_count | numeric | Number of low operational risk. |
| operational_ok_count | numeric | Number of no operational risk. |
| origin_id | text | Origin ID.<br>Note that origin ID is blank if the component does not have a distribution. |
| origin_name | text | Name of the distribution (origin). |
| policy_approval_status | text | One of the following values:<br>• IN_VIOLATION<br>• NOT_IN_VIOLATION<br>• IN_VIOLATION_OVERRIDDEN |

| Column | Type | Description |
|---|---|---|
| project_version_id | UUID | Project version ID. |
| purl | text | The package URL. |
| review_status | text | Review status of the component. Possible values are:<br>• UNREVIEWED<br>• IN_REVIEW<br>• REVIEWED<br>• APPROVED<br>• LIMITED_APPROVAL<br>• REJECTED<br>• DEPRECATED |
| reviewed_by | UUID | User who reviewed the component. |
| reviewed_on | timestamp in UTC | When the component was reviewed. |
| security_critical_count | numeric | Number of critical security vulnerabilities. |
| security_high_count | numeric | Number of high security vulnerabilities. |
| security_medium_count | numeric | Number of medium security vulnerabilities. |
| security_low_count | numeric | Number of low security vulnerabilities. |
| security_ok_count | numeric | Number of no security vulnerabilities. |
| updated_at | timestamptz | The updated at time for the component, copied from the BOM.<br>Represents the most recent time that component was updated in its BOM. |
| version_origin_id | UUID | Version origin ID. |

# Component table (Ignored components only) (component_ignored)

| Column | Type | Description |
|---|---|---|
| component_id | UUID | Component ID. |
| component_name | text | Component name. |
| component_version_id | UUID | Component version ID. |
| component_version_name | text | Component version name. |
| id | int8 | ID. |
| ignored | boolean | Indicates whether the component is ignored:<br>• "t" indicates that the component is ignored.<br>• "f" indicates that the component is not ignored. |

| Column | Type | Description |
|---|---|---|
| license_high_count | numeric | Number of high license risk. |
| license_medium_count | numeric | Number of medium license risk. |
| license_low_count | numeric | Number of low license risk. |
| license_ok_count | numeric | Number of no license risk. |
| operational_high_count | numeric | Number of high operational risk. |
| operational_medium_count | numeric | Number of medium operation risk. |
| operational_low_count | numeric | Number of low operational risk. |
| operational_ok_count | numeric | Number of no operational risk. |
| origin_id | text | Origin ID.<br>Note that origin ID is blank if the component does not have a distribution. |
| origin_name | text | Name of the distribution (origin). |
| policy_approval_status | text | One of the following values:<br>• IN_VIOLATION<br>• NOT_IN_VIOLATION<br>• IN_VIOLATION_OVERRIDDEN |
| project_version_id | UUID | Project version ID. |
| review_status | text | Review status of the component. Possible values are:<br>• UNREVIEWED<br>• IN_REVIEW<br>• REVIEWED<br>• APPROVED<br>• LIMITED_APPROVAL<br>• REJECTED<br>• DEPRECATED |
| reviewed_by | UUID | User who reviewed the component. |
| reviewed_on | timestamp in UTC | When the component was reviewed. |
| security_critical_count | numeric | Number of critical security vulnerabilities. |
| security_high_count | numeric | Number of high security vulnerabilities. |
| security_medium_count | numeric | Number of medium security vulnerabilities. |
| security_low_count | numeric | Number of low security vulnerabilities. |
| security_ok_count | numeric | Number of no security vulnerabilities. |
| version_origin_id | UUID | Version origin ID. |

## Component Comments table (component_comments)

| Column | Type | Description |
|---|---|---|
| comment | text | Text of the comment. |
| comment_id | UUID | ID of the comment. |
| component_table_id | int8 | ID of the component in the reporting.component table containing the comment. |
| created_at | timestamp in UTC | When the comment was created. |
| created_by | UUID | User who created the comment |
| project_id | UUID | Project ID of the project containing the comment. |
| project_version_id | UUID | Project version ID of the project version where this BOM component appears in the BOM. |
| updated_at | timestamp in UTC | When the comment was last updated. |

## Component Custom Fields table (component_custom_fields)

| Column | Type | Description |
|---|---|---|
| active | boolean | Defines whether this custom field is active.<br>• "true" indicates the custom field is active.<br>• "false" indicates the custom field is deactivated. |
| component_id | int8 | ID of the BOM component.<br>Use this column to join with other reporting.component* tables to obtain more information. |
| custom_field_id | integer | ID of the BOM component custom field. |
| custom_field_label | text | Label of this custom field. |
| custom_field_type | text | Type of custom field. For example, MULTISELECT or TEXT. |
| project_version_id | UUID | Project version ID of the project version where this BOM component appears in the BOM. |
| values | text[] | Data stored for this BOM component custom field for this component. |

## Component License table (component_license)

| Column | Type | Description |
|---|---|---|
| component_table_id | int8 | id field in the Component table. |
| id | int8 | ID. |
| license_display | text | License name when it is a single license; license display when it is a complex license. For example, (License A OR license B). |
| license_family_name | text | License family this license belongs to for purposes of risk calculations and the definition of open source policy rules. |
| project_version_id | UUID | Project version ID. |

## Component Match Type table (component_match_types)

| Column | Type | Description |
|---|---|---|
| component_id | int8 | id field in the Component table. |
| match_type | text | One of the following values:<br>• BINARY<br>• FILE_FILES_ADDED_DELETED_AND_MODIFIED<br>• FILE_DEPENDENCY<br>• FILE_DEPENDENCY_DIRECT<br>• FILE_DEPENDENCY_TRANSITIVE<br>• FILE_EXACT<br>• FILE_EXACT_FILE_MATCH<br>• FILE_SOME_FILES_MODIFIED<br>• MANUAL_BOM_COMPONENT<br>• MANUAL_BOM_FILE<br>• PARTIAL_FILE<br>• SNIPPET |
| project_version_id | UUID | Project version ID. |

## Component Matches table (component_matches)

| Column | Type | Description |
|---|---|---|
| component_table_id | int8 | id field in the Component table. |
| match_archive_context | text | Local path to the archived file relative to the project's root directory. |
| match_confidence | numeric | Represents the confidence in the match, excluding, snippet, binary, or partial file matches. |

| Column | Type | Description |
|---|---|---|
| | | For manually added components and components added via package manager scans, the value is always 100%. Components found via signature scans are the only components that have a confidence value between 0.00 (no confidence) and 100.00 (confident). |
| match_file_name | text | File name |
| match_id | int8 | Match ID. |
| match_path | text | Path. |
| match_type | text | One of the following values:<br>• BINARY<br>• FILE_FILES_ADDED_DELETED_AND_MODIFIED<br>• FILE_DEPENDENCY<br>• FILE_DEPENDENCY_DIRECT<br>• FILE_DEPENDENCY_TRANSITIVE<br>• FILE_EXACT<br>• FILE_EXACT_FILE_MATCH<br>• FILE_SOME_FILES_MODIFIED<br>• MANUAL_BOM_COMPONENT<br>• MANUAL_BOM_FILE<br>• PARTIAL_FILE<br>• SNIPPET |
| project_version_id | UUID | Project version ID. |
| snippet_confirmation_status | text | Review status of the snippet matches. Possible values are:<br>• Reviewed<br>• Not Reviewed<br>• Null<br>Note that snippet matches that have not been reviewed will not appear in the reporting database. |

# Component Policies table (component_policies)

| Column | Type | Description |
|---|---|---|
| category | text | Policy Category information. Current values are:<br>• COMPONENT<br>• LICENSE<br>• OPERATIONAL<br>• SECURITY<br>• UNCATEGORIZED |

| Column | Type | Description |
|---|---|---|
| component_table_id | int8 | ID field in the Component table. |
| description | text | Policy description. |
| overridden_at | timestamp with time zone | When the policy was overridden. |
| overridden_by | UUID | User who overrode the policy. |
| override_comment | text[] | Notes about this version of the project. |
| policy_id | UUID | Policy ID. |
| policy_name | text | Name of the policy. |
| policy_status | text | Status of the policy. |
| project_version_id | UUID | Project version ID. |
| severity | text | Severity level of the policy. Possible values are:<br>• BLOCKER<br>• CRITICAL<br>• MAJOR<br>• MINOR<br>• TRIVIAL |

## Component Usage table (component_usages)

| Column | Type | Description |
|---|---|---|
| component_id | int8 | id field in the Component table. |
| project_version_id | UUID | ID. |
| usage | text | One of the following values:<br>• DYNAMICALLY_LINKED<br>• STATICALLY_LINKED<br>• SOURCE_CODE<br>• DEV_TOOL_EXCLUDED<br>• SEPARATE_WORK<br>• IMPLEMENTATION_OF_STANDARD |

## Component Vulnerability table (component_vulnerability)

| Column | Type | Description |
|---|---|---|
| actual_date | timestamp with time zone | Actual date the vulnerability was remediated. |
| attack_vector | text | Attack vector of the vulnerability, which is the context by which vulnerability exploitation is possible. Possible values are: |

| Column | Type | Description |
|---|---|---|
| | | • NETWORK<br>• ADJACENT<br>• LOCAL<br>• PHYSICAL |
| base_score | numeric | Base score of the vulnerability based on the CVSS v2 score. This score reflects the overall basic characteristics of a vulnerability that are constant over time and user environments. |
| base_score_cvss3 | numeric | Base score of the vulnerability based on the CVSS v3.x score. This score reflects the overall basic characteristics of a vulnerability that are constant over time and user environments. |
| comment | text | Comments entered when remediating the vulnerability. |
| component_table_id | int8 | ID field in the Component table. |
| cwe_ids | text | List of Common Weakness Enumeration (CWE) IDs for this security vulnerability. |
| description | text | Description of the vulnerability. |
| exploit_score | numeric | Exploitability score of the vulnerability based on the CVSS v2 score. This score measures how the vulnerability is accessed and if extra conditions are required to exploit it, taking into account access vector, complexity, and authentication. |
| exploit_score_cvss3 | numeric | Exploitability score of the vulnerability based on the CVSS v3.x score. This score measures how the vulnerability is accessed and if extra conditions are required to exploit it, taking into account access vector, complexity, and authentication. |
| exposed_on | timestamp with time zone | When the vulnerability was mapped to the project. |
| impact_score | numeric | Impact score of the vulnerability based on the CVSS v2 score. This score reflects the possible impact of successfully exploiting the vulnerability, considering the integrity, availability, and confidentiality impacts. |
| impact_score_cvss3 | numeric | Impact score of the vulnerability based on the CVSS v3.x score. This score reflects the possible impact of successfully exploiting the vulnerability, considering the integrity, availability, and confidentiality impacts. |
| project_version_id | UUID | ID. |

| Column | Type | Description |
|---|---|---|
| published_on | timestamp with time zone | When the vulnerability was published. |
| related_vuln_id | text | Empty except when BDSA has a related CVE vulnerability. If a BDSA vulnerability is mapped to a CVE, the related CVE is listed here; the BDSA vulnerability is listed in the vuln_id column. |
| remediation_status | text | Lists the remediation status. One of the following values:<br>• NEW<br>• NEEDS_REVIEW<br>• REMEDIATION_REQUIRED<br>• REMEDIATION_COMPLETE<br>• DUPLICATE<br>• MITIGATED<br>• PATCHED<br>• IGNORED |
| remediation_updated_at | timestamp with time zone | When the triage status was updated for this vulnerability. |
| severity | text | Severity level of this vulnerability based on the CVSS v2 score. One of the following values:<br>• HIGH<br>• MEDIUM<br>• LOW |
| severity_cvss3 | text | Severity level of this vulnerability based on the CVSS v3.x score. One of the following values:<br>• CRITICAL<br>• HIGH<br>• MEDIUM<br>• LOW |
| solution_available | boolean | Indicates whether a solution for the vulnerability is available:<br>• "t" indicates a solution is available.<br>• "f" indicates a solution is not available. |
| target_date | timestamp with time zone | Target date to remediate the vulnerability |
| temporal_score | numeric | Temporal score of the vulnerability based on the CVSS v2 score. This score represents time-dependent qualities of a vulnerability, taking into account the confirmation of the technical details of a vulnerability, the existence of any patches or workarounds, and the availability of exploit code or techniques. Displays 0 if there is no score. |

| Column | Type | Description |
|---|---|---|
| temporal_score_cvss3 | numeric | Temporal score of the vulnerability based on the CVSS v3.x score. This score represents time-dependent qualities of a vulnerability, taking into account the confirmation of the technical details of a vulnerability, the existence of any patches or workarounds, and the availability of exploit code or techniques. Displays 0 if there is no score. |
| updated_on | timestamp with time zone | When the vulnerability was last updated. |
| vuln_id | text | Vulnerability ID, such as CVE-2017-1234 or 12345. |
| vuln_source | text | Source of the vulnerability. One of the following values:<br>• BDSA<br>• NVD |
| workaround_available | boolean | Indicates whether a workaround for the vulnerability is available:<br>• "t" indicates a workaround is available.<br>• "f" indicates a workaround is not available. |

# Project table (project)

| Column | Type | Description |
|---|---|---|
| created_at | timestamp with time zone | Project creation date. |
| description | text | Project description. |
| owner | UUID | User ID in Black Duck. |
| project_id | UUID | Project ID |
| project_name | text | Project name. |
| tier | smallint | Project tier. A value between 0 - 5. |

# Project Custom Fields table

| Column | Type | Description |
|---|---|---|
| active | boolean | Defines whether this custom field is active.<br>• "true" indicates the custom field is active.<br>• "false" indicates the custom field is deactivated. |

| Column | Type | Description |
|---|---|---|
| custom_field_id | integer | ID of the project custom field. |
| custom_field_label | text | Label of this custom field. |
| custom_field_type | text | Type of this custom field. For example, MULTISELECT or TEXT. |
| project_id | UUID | UUID of the project where this project custom field appears. |
| values | text[] | Data stored for this project custom field for this project. |

## Project Mapping table (project_mapping)

| Column | Type | Description |
|---|---|---|
| application_id | text | Application ID. |
| project_id | UUID | Project ID. |

## Project Version table (project_version)

| Column | Type | Description |
|---|---|---|
| created_on | timestamp with time zone | Project version creation date. |
| distribution | text | Project Distribution:<br>• EXTERNAL<br>• SAAS<br>• INTERNAL<br>• OPENSOURCE |
| license_high_component_count | integer | Number of component origins in this project version that have at least one high license risk. |
| license_medium_component_count | integer | Number of component origins in this project version that have at least one medium license risk. |
| license_low_component_count | integer | Number of component origins in this project version that have at least one low license risk. |
| license_ok_component_count | integer | Number of component origins in this project version that have no license risk. |
| nickname | text | Nickname for the project version. |
| notes | text | Notes about this version of the project. |

| Column | Type | Description |
|---|---|---|
| operational_high_component_count | integer | Number of component origins in this project version that have at least one high operational risk. |
| operational_medium_component_count | integer | Number of component origins in this project version that have at least one medium operational risk. |
| operational_low_component_count | integer | Number of components in this project version that have at least one low operational risk. |
| operational_ok_component_count | integer | Number of components in this project version that have no operational risk. |
| phase | text | Project phase:<br>• PLANNING<br>• DEVELOPMENT<br>• PRERELEASE<br>• RELEASED<br>• DEPRECATED<br>• ARCHIVED |
| project_id | UUID | Project ID. |
| released_on | timestamp with time zone | Project release date. |
| security_critical_component_count | integer | Number of component origins in this project version that have at least one critical security risk. |
| security_high_component_count | integer | Number of component origins in this project version that have at least one high security risk. |
| security_medium_component_count | integer | Number of component origins in this project version that have at least one medium security risk. |
| security_low_component_count | integer | Number of component origins in this project version that have at least one low security risk. |
| security_ok_component_count | integer | Number of component origins in this project version that have no security risk. |
| updated_at | timestamp with time zone | When the project version was last updated. |
| version_id | UUID | Project version ID. |
| version_name | text | Project version name. |

## Project Version Code Location table (project_version_code_location)

| Column | Type | Description |
|---|---|---|
| id | UUID | ID. |
| last_scan_time | int8 | Time of last scan. |
| name | text | Code location name |
| project_version_id | UUID | Project version ID. |

## Project Version Custom Fields table (project_version_custom_fields)

| Column | Type | Description |
|---|---|---|
| active | boolean | Defines whether this custom field is active.<br>• "true" indicates the custom field is active.<br>• "false" indicates the custom field is deactivated. |
| custom_field_id | integer | ID of the custom field. |
| custom_field_label | text | Label of this custom field. |
| custom_field_type | text | Type of custom field. For example, MULTISELECT or TEXT. |
| project_version_id | UUID | UUID of the project version where this custom field appears. |
| values | text | Data stored for this project version custom field for this project. |

## Rapid Scan aggregate stats view table (scan_rapid_aggregate_stats_view)

| Column | Type | Description |
|---|---|---|
| avg_duration_ms | numeric | The average duration in milliseconds over the sample time period. |
| avg_num_components | numeric | The average number of identified components. |
| avg_num_components_with_policy_bloc | numeric | The average number of identified components with blocker severity policy rule violations. |

2. Report Database Schema • Rapid Scan aggregate stats view table (`scan_rapid_aggregate_stats_view`)

| Column | Type | Description |
|---|---|---|
| `avg_num_components_with_policy_crit` | numeric | The average number of identified components with critical severity policy rule violations. |
| `avg_num_components_with_policy_major` | numeric | The average number of identified components with major severity policy rule violations. |
| `avg_num_components_with_policy_minor` | numeric | The average number of identified components with minor severity policy rule violations. |
| `avg_num_components_with_policy_triv` | numeric | The average number of identified components with trivial severity policy rule violations. |
| `avg_num_components_with_policy_unsp` | numeric | The average number of identified components with unspecified severity policy rule violations. |
| `avg_num_components_with_vuln_crit` | numeric | The average number of identified components with critical severity vulnerabilities. |
| `avg_num_components_with_vuln_high` | numeric | The average number of identified components with high severity vulnerabilities. |
| `avg_num_components_with_vuln_low` | numeric | The average number of identified components with low severity vulnerabilities. |
| `avg_num_components_with_vuln_med` | numeric | The average number of identified components with medium severity vulnerabilities. |
| `avg_num_components_with_vuln_none` | numeric | The average number of identified components with none severity vulnerabilities. |
| `avg_num_licenses_internal_prop` | numeric | The average number of identified licenses with internal proprietary license families. |
| `avg_num_licenses_permissive` | numeric | The average number of identified licenses with permissive license families. |
| `avg_num_licenses_reciprocal` | numeric | The average number of identified licenses with reciprocal license families. |
| `avg_num_licenses_reciprocal_agpl` | numeric | The average number of identified licenses with reciprocal AGPL license families. |
| `avg_num_licenses_restricted_prop` | numeric | The average number of identified licenses with restricted proprietary license families. |

2. Report Database Schema • Rapid Scan aggregate stats view table (`scan_rapid_aggregate_stats_view`)

| Column | Type | Description |
|---|---|---|
| `avg_num_licenses_unknown` | numeric | The average number of identified licenses with unknown license families. |
| `avg_num_licenses_weak_reciprocal` | numeric | The average number of identified licenses with weak reciprocal license families. |
| `avg_num_policies_by_blocker` | numeric | The average number of blocker severity policy rule violations. |
| `avg_num_policies_by_crit` | numeric | The average number of critical severity policy rule violations. |
| `avg_num_policies_by_major` | numeric | The average number of major severity policy rule violations. |
| `avg_num_policies_by_minor` | numeric | The average number of minor severity policy rule violations. |
| `avg_num_policies_by_trivial` | numeric | The average number of trivial severity policy rule violations. |
| `avg_num_policies_by_unspecified` | numeric | The average number of unspecified severity policy rule violations. |
| `avg_num_scans_per_project` | numeric | The average number of scans per project. |
| `avg_num_scans_per_user` | numeric | The average number of scans per user. |
| `avg_num_scans_per_version` | numeric | The average number of scans per project version. |
| `avg_num_vuln_count_by_critical` | numeric | The average number of critical severity vulnerabilities. |
| `avg_num_vuln_count_by_high` | numeric | The average number of high severity vulnerabilities. |
| `avg_num_vuln_count_by_low` | numeric | The average number of low severity vulnerabilities. |
| `avg_num_vuln_count_by_med` | numeric | The average number of medium severity vulnerabilities. |
| `avg_num_vuln_count_by_none` | numeric | The average number of none severity vulnerabilities. |
| `distinct_projects` | int4 | The count of unique projects. |
| `distinct_users` | int4 | The count of unique users. |
| `distinct_versions` | int4 | The count of unique project versions. |
| `end_time` | timestamp with time zone | The ending timestamp of this sample time period. |
| `iqr_duration_ms` | int4 | The interquartile range of scan times over the sample time period. |

| Column | Type | Description |
|---|---|---|
| max_duration_ms | int4 | The maximum duration in milliseconds over the sample time period. |
| max_num_components | int4 | The maximum number of identified components. |
| min_duration_ms | int4 | The minimum duration in milliseconds over the sample time period. |
| sample_time | timestamp with time zone | The beginning timestamp of this sample time period. |
| stddev_duration_ms | numeric | The standard deviation duration in milliseconds over the sample time period. |
| total_components | int4 | The total number of components. |
| total_components_with_pol_viol | int4 | The total number of components with policy rule violations. |
| total_noproject_scans | int4 | The total number of scans without a project. |
| total_noversion_scans | int4 | The total number of scans without a project version. |
| total_num_scans | int4 | The total number of scans over the sample time period. |
| total_q1_scans | int4 | The total number of scans considered ‘short scans’. |
| total_q4_scans | int4 | The total number of scans considered ‘long scans’. |

## Scan stats view table (scan_stats_view)

| Column | Type | Description |
|---|---|---|
| application_id | text | The external application id that is associated to the mapped project. |
| code_location_id | UUID | The code location ID. |
| code_location_name | text | The code location name. |
| parent_project_group_id | UUID | The parent project group ID to which the scan is associated. |
| project_group_id | UUID | The project group ID to which the scan is associated. |
| project_group_name | text | The project group name to which the scan is associated. |
| project_id | UUID | The project ID to which the scan is associated. |

| Column | Type | Description |
|---|---|---|
| project_name | text | The project name to which the scan is associated. |
| scan_age | interval | The age since the scan was created. |
| scan_archived_at | timestamp | The timestamp at which the scan was archived. |
| scan_duration | interval | The scan duration. |
| scan_end_at | timestamp with time zone | The timestamp at which the scan ended. |
| scan_id | UUID | The scan ID. |
| scan_name | text | The scan name. |
| scan_size | int8 | The file system size in bytes of inspected uncompressed files in the scan. |
| scan_start_at | timestamp with time zone | The timestamp at which the scan started. |
| scan_status | varchar | The transition reason for the scan. |
| scan_type | text | The scan type. |
| uploaded_at | timestamp with time zone | The timestamp at which the scan data was uploaded from the client. |
| user_id | UUID | The user ID. |
| version_id | UUID | The project version ID to which the scan is associated. |
| version_name | text | The project version name to which to the scan is associated. |

## Scan view table (scan_view)

| Column | Type | Description |
|---|---|---|
| application_id | text | The external application id that is associated to the mapped project. |
| archive_initiated_by | UUID | The user that archived the scan. |
| archived_at | timestamp | The timestamp at which the scan was archived. |
| basedir | text | The base directory path under which the scan occurred (signature scans). |
| code_location_id | UUID | The code location ID. |
| code_location_name | text | The code location name. |
| created_by_user_id | UUID | The user ID of the user that created the scan. |
| file_system_size | int8 | The file system size in bytes of inspected uncompressed files in the scan. |

| Column | Type | Description |
|---|---|---|
| host_name | varchar | The host name under which the scan occurred (signature scans). |
| match_count | int4 | The number of identified matches from the scan. |
| name | text | The scan name. |
| num_dirs | int4 | The number of directories inspected within the scan. |
| num_non_dir_files | int4 | The number of non-directory files inspected within the scan. |
| project_id | UUID | The project ID to which the scan is associated. |
| project_name | text | The project name to which the scan is associated. |
| scan_end_at | timestamp with time zone | The timestamp at which the scan ended. |
| scan_id | UUID | The scan ID. |
| scan_source_id | text | The internal scan source ID. |
| scan_source_type | varchar | The internal scan source type. |
| scan_start_at | timestamp with time zone | The timestamp at which the scan started. |
| scantime | int8 | The timestamp at which the scan occurred (millis since epoch). |
| server_version | varchar | The Black Duck server version under which this scan occurred. |
| status | varchar | The transition reason for the scan. |
| status_message | text | The message associated with the scan's transition reason. |
| timelastmodified | int8 | The timestamp at which the scan was last modified (millis since epoch). |
| type | text | The scan type. |
| uploaded_at | timestamp with time zone | The timestamp at which the scan data was uploaded from the client. |
| version_id | UUID | The project version id to which the scan is associated. |
| version_name | text | The project version name to which the scan is associated. |

# User table (user)

| Column | Type | Description |
|---|---|---|
| active | boolean | Defines whether this user is active. |

| Column | Type | Description |
|---|---|---|
| | | • "true" indicates this user is active.<br>• "false" indicates this user is inactive. |
| email | text | User's email address. |
| first_name | text | User's first name. |
| id | UUID | ID. |
| last_login | timestamp with timezone | Time that the user last logged in to Black Duck. |
| last_name | text | User's last name. |
| username | text | User's username in Black Duck. |

## User group project mapping table (user_group_project_mapping)

| Column | Type | Description |
|---|---|---|
| group_id | UUID | The user group ID. |
| group_name | text | The user group name. |
| project_id | UUID | The project ID to which this user group is mapped. |
| project_name | text | The project name to which this user group is mapped. |
| user_id | UUID | The user ID that is a member of this user group. |
| user_name | text | The user name of the user ID above. |

## Vulnerability Method Matches table (vulnerability_method_matches)

| Column | Type | Description |
|---|---|---|
| called_function | text | Name of the vulnerable function call in your code that makes the vulnerability reachable. |
| id | int8 | ID. |
| line_number | integer | Line number in your code where the vulnerable function is called. |
| project_version_id | UUID | UUID of the project version where the reachable vulnerability appears. |
| qualified_name | text | Name of the class the function is called on. |
| vuln_id | text | Vulnerability ID, such as BDSA-2020-1234. |

2. Report Database Schema • Vulnerability Method Matches table (`vulnerability_method_matches`)

| Column | Type | Description |
|---|---|---|
| `vuln_source` | text | Source of the vulnerability. For vulnerability impact analysis, the value is BDSA. |