Preface

This guide provides a single location with everything you need to know about configuring and using SQL Firewall.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Documents

For more security-related information, see these Oracle resources:

- *Oracle Database Security Guide*
- *Oracle Database SQL Language Reference*
- *Oracle Database Reference*
- *Oracle Data Guard Concepts and Administration*
- *Oracle Database PL/SQL Packages and Types Reference*
- *Oracle Database Utilities*
- *Oracle Audit Vault and Database Firewall Auditor's Guide*
- *Using Oracle Data Safe*

Many of the examples in this guide use the sample schemas of the seed PDB, which you can create when you install Oracle Database. See *Oracle Database Sample Schemas* for information about how these schemas were created and how you can use them yourself.

Oracle Technical Services

To download the product data sheet, frequently asked questions, links to the latest product documentation, product download, and other collateral, visit Oracle Technical Resources (formerly Oracle Technology Network). You must register online before using Oracle Technical Services. Registration is free and can be done at

https://www.oracle.com/technical-resources/

My Oracle Support

You can find information about security patches, certifications, and the support knowledge base by visiting My Oracle Support (formerly Oracle*MetaLink*) at

https://support.oracle.com

Changes in This Release for Oracle Database SQL Firewall Guide

This preface contains:

Changes in Oracle Database SQL Firewall 23ai

Oracle Database SQL Firewall Guide for Oracle Database 23ai has new security features.

Oracle SQL Firewall is Now Built into Oracle Database

Starting with Oracle Database 23ai, Oracle SQL Firewall inspects all incoming SQL statements and ensures that only explicitly authorized SQL is run.

You can use SQL Firewall to control which SQL statements are allowed to be processed by the database. You can restrict connection paths associated with database connections and SQL statements. Unauthorized SQL can be logged and blocked.

Because SQL Firewall is built into the Oracle database, it cannot be bypassed. All SQL statements are inspected, whether local or network, or encrypted or clear text. It examines top-level SQL, stored procedures and the related database objects.

SQL Firewall provides real-time protection against common database attacks by restricting database access to only authorized SQL statements or connections. It mitigates risks from SQL injection attacks, anomalous access, and credential theft or abuse.

SQL Firewall uses session context data such as IP address, operating system user name, and operating system program name to restrict how a database account can connect to the database. This helps mitigate the risk of stolen or misused application service account credentials. A typical use case for SQL Firewall is for application workloads.

You can use SQL Firewall in both the root and a pluggable database (PDB).

Updates to Oracle Database SQL Firewall 23ai

Oracle Database SQL Firewall Guide for Oracle Database 23ai as the following update.

New Procedure for Oracle SQL Firewall DBMS_SQL_FIREWALL PL/SQL Package

The Oracle SQL Firewall package `DBMS_SQL_FIREWALL` now has an additional procedure, `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL`.

This procedure enables you to individually append specific SQL records from a capture log or a violation log to an existing allow-list. While `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST` provides the flexibility to append the entire violation or capture log to the allow-list, in most common

scenarios you might also need the flexibility to add just one of them instead of the entire list. In previous releases, if you wanted to append specific SQL commands to an allow-list, you had to use `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST` to append the entire violation or capture log to the allow-list, and then use `DBMS_SQL_FIREWALL.DELETE_ALLOWED_LIST` to manually delete the unwanted entries. This enhancement gives more flexibility to adjust the allow-list with specific records that you want to include.

Related Topics

- *Oracle Database PL/SQL Packages and Types Reference*

1

Overview of Oracle SQL Firewall

SQL Firewall is part of the Oracle Database kernel. Learn about Oracle SQL Firewall and its use cases and features from this section.

1.1 About Oracle SQL Firewall

Oracle SQL Firewall provides real-time protection against common database attacks by restricting database access to only authorized SQL statements or connections for a designated user.

It mitigates risks from SQL injection attacks, anomalous access, and credential theft or abuse, preventing or detecting potential SQL injection attacks.

You can use SQL Firewall to control which SQL statements are allowed to be processed by the database. In addition, SQL Firewall can use session context data such as IP address to restrict database connections. Unauthorized SQL and database connection can be logged and blocked.

SQL Firewall helps to address the following three **use cases**:

- Provide real-time protection by restricting database access to only authorized SQL statements and database connections.
- Mitigate SQL injection attacks, anomalous access, and credential theft/abuse risks.
- Enforce trusted database connection paths.

SQL Firewall offers the following **benefits**:

- SQL Firewall inspects all incoming database connections and SQL statements, including those from PL/SQL, whether local or over the network, encrypted or clear text. It cannot be bypassed. It only allows explicitly authorized SQL. For all other SQL, it logs the offending statements and raises violations. This statement could have been a SQL injection attack or a new SQL statement that the authorized user has not run before.
- You can decide whether you want to block unauthorized SQL or only log it. This gives you the flexibility on how to handle attacks.
- SQL Firewall evaluates the complete SQL and the processing context. By running inside the Oracle database server, the firewall easily handles encoding of the SQL statement, synonyms, dynamically generated object names, and any SQL statements that are dynamically generated in PL/SQL units.
- SQL Firewall relies on the allow-listing (an allow-list is a set of permitted actions) of the authorized SQL statements and associated trusted database connection paths while blocking the rest. You train the SQL Firewall by simply capturing authorized SQL statements for a database account. Subsequently, the firewall detects and prevents unauthorized SQL and potential SQL injection attacks. A typical use case with allow-listed SQL statements is for application SQL workloads issued by application service account.
- SQL Firewall can also block connections that do not come from trusted IP addresses, operating system user names, or program names. This function is useful when you want to put some protection in place immediately, while you create the allow-list of SQL statements for your applications. This feature ensures that any direct access to your databases is

coming exclusively from trusted endpoints. This also helps mitigate the risk of stolen or misused application service account credentials.

SQL Firewall enables you to build an allow-list policy for each database user of SQL statements that a typical database user performs, and then detects, blocks, and logs any unexpected SQL.

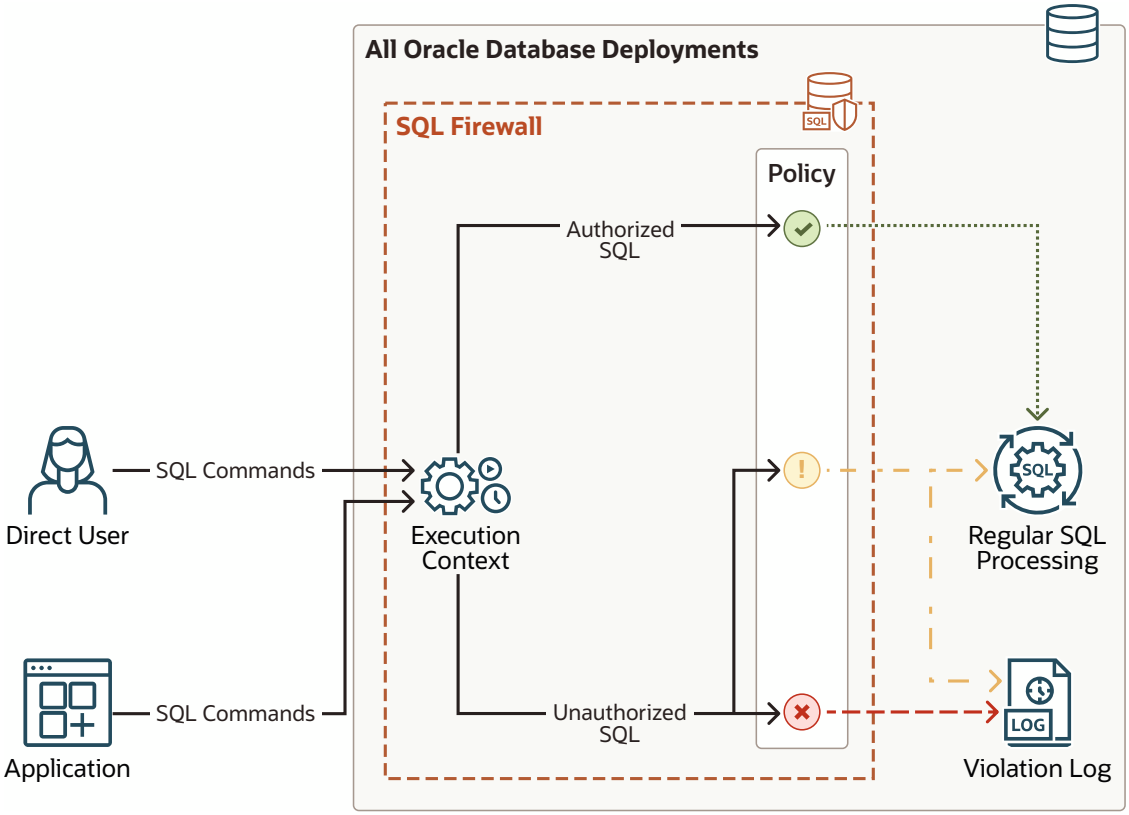
SQL Firewall policies work at a database account level, whether of an application service account or a direct database user, such as a reporting user or a database administrator. In other words, you might have one SQL Firewall policy for the database user `HR` and another for the database user `pfitch`. This flexibility allows you to gradually build up the protection level of the database, starting from either the database administrators or the application service accounts.

You can use SQL Firewall in both the root and a pluggable database (PDB). SQL Firewall is a simple and easy-to-use firewall solution for all Oracle Database deployments, such as on-premises, cloud, multitenant, Oracle Data Guard, or Oracle Real Application Clusters. SQL Firewall works in conjunction with other Oracle Database security features such as Transparent Data Encryption (TDE), database auditing, and Oracle Database Vault.

SQL Firewall supports (that is, it captures and enforces on) all SQL commands except transaction control commands (`SAVEPOINT`, `COMMIT`, and `ROLLBACK`). Additionally, SQL Firewall supports the SQL*Plus commands `PASSWORD` and `DESCRIBE`, and remote procedure calls (RPC) through database links.

The following diagram explains how SQL Firewall operates inline within the Oracle Database kernel.

Figure 1-1 SQL Firewall Process



1. A user logs in to the Oracle database through a web application.
2. The user runs SQL statements, creating inbound traffic to the Oracle database.
3. SQL Firewall inspects the incoming database connections and SQL statements, and enforces it against the permitted SQL statements and trusted connection paths in the allow-list policy for the user. SQL Firewall's processing outcome is one of the following options:
   - Allow the SQL for its subsequent execution.
   - Allow the SQL and log it.
   - Log and optionally block unauthorized SQL.

1.2 Licensing Oracle SQL Firewall

Oracle SQL Firewall must be licensed for use. There are two paths to its license.

- **Included with Oracle Database Vault.** Oracle Database Vault is an extra-cost option of Oracle Database. See *Oracle Database Licensing Information User Manual*.
- **Included with Oracle Audit Vault and Database Firewall (AVDF).** AVDF is a separate Oracle product and requires a license. See *Oracle Database Licensing Information User Manual*.

1.3 Getting Started with Oracle SQL Firewall

To get started with Oracle SQL Firewall, you follow three steps: first, enable Oracle SQL Firewall; second, capture the user's normal SQL activities; and third, enable and enforce allow-lists.

1. **Enable SQL Firewall.** As an administrator with appropriate privileges, enable SQL Firewall in the Oracle database.
2. **Capture the normal SQL activities.** For every database user that you want to protect with SQL Firewall, you must enable SQL Firewall to learn the normal SQL traffic of the database user. It does this by capturing all the authorized SQL statements over trusted database connection paths. You can query SQL Firewall-specific data dictionary views to review this captured data to determine if the collected SQL statements and connection paths is adequate to constitute the allow-lists.
   After you review the captured SQL statements, you can generate a SQL Firewall policy with allow-lists that set the baseline for allowed SQL statements and allowed contexts. Allowed SQL statements constitute the approved SQL statements. At run-time, when the policy is enforced, any incoming SQL queries that have a structure syntactically similar to the SQL signature in the policy allow-list will be passed for execution if the corresponding run-time execution context also meets the set of allowed contexts. Allowed contexts represent trusted database connection paths and consist of three distinct groups—client IP addresses, operating system program names, and operating system user names. When the user connects to the database, SQL Firewall checks the current session context attributes, and ensures that access to the database comes exclusively from trusted endpoints defined in the allow-lists. You can review the allow-list and make modifications by using the `DBMS_SQL_FIREWALL` procedures any time.
3. **Enable and enforce the allow-lists.** Enabling the generated SQL Firewall policy protects the database user. SQL Firewall enforces and checks the allow-lists when the user connects to the database and issues SQL statements. You can let SQL Firewall know if you want to enforce checks on allowed contexts, allowed SQL statements, or both. If the database connection paths and SQL statements in the incoming SQL traffic do not match

the entries in the enabled and enforced allow-lists, then a SQL Firewall violation is triggered and this incident is logged in the violation log. You can let SQL Firewall know how to respond to SQL Firewall violation incident: allow the traffic to proceed to the database or block. Blocking raises an `ORA-47605: SQL Firewall violation` error, which prevents anomalous database access, without disrupting client connections for SQL violations following a mismatch of SQL statements. However, blocking for context violations will disrupt and terminate client connections following a mismatch of contexts.
SQL Firewall raises and logs violations in real-time for every unmatched scenario of database connection or SQL command execution against the entries in the enabled allow-lists of the SQL Firewall policy. A security administrator can monitor the SQL Firewall violation log `DBA_SQL_FIREWALL_VIOLATIONS` to detect the presence of these abnormalities. You may want to audit SQL Firewall violations (especially the blocked ones); their occurrence potentially indicates abnormal database access attempts including SQL Injection and credential theft or abuse. Auditing violations places a record of the violation in the database audit trail, where it can be protected from tampering.

Key points to consider are as follows:

- Oracle Database mandatorily audits all SQL Firewall administrative actions and writes these to the unified audit trail data dictionary view, `UNIFIED_AUDIT_TRAIL`. You can also create unified audit policies to monitor SQL Firewall violations. Another way to monitor and troubleshoot SQL Firewall is to use the `SQL_FIREWALL` trace file setting.
- You can export and import SQL Firewall metadata, including existing allow-lists, by using the Oracle Data Pump `EXPDB` and `IMPDB` utilities.
- Oracle recommends that you periodically monitor and purge violations logs by using the `DBMS_SQL_FIREWALL.PURGE_LOG` procedure as part of routine SQL Firewall management tasks. In a well trained environment, violation logs are not expected to be voluminous.
- SQL Firewall captures SQL statements that the user issues directly or from PL/SQL units that the user invokes in sessions of target users.
- SQL Firewall captures only SQL statements that are executed successfully. That is, if a SQL statement fails to execute due to any error, SQL Firewall does not capture the corresponding statement.
- SQL Firewall captures SQL statements before any internal query transformation (for example, views or macro expansions, or Oracle Virtual Private Database policy enforcement) is performed.
- SQL Firewall normalizes captured SQL statements and replaces literal values with special symbols before storing them in the log tables.
- The session context attributes (client IP address, operating system user name, and operating system program name) are checked only once during session creation.
- You can append to the existing allow-list anytime by using either the `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST` procedure or the `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL` procedure from the following two sources:
  - Violation log: `DBA_SQL_FIREWALL_VIOLATIONS` data dictionary view
  - Capture log: `DBA_SQL_FIREWALL_CAPTURE_LOGS` data dictionary view
- For existing sessions that were created before the allow-list is enabled, SQL Firewall also checks the allowed contexts, but does not terminate existing sessions even if they have unmatched session contexts. In this case, SQL Firewall does not log the violation.

1.4 Privileges for Configuring and Using Oracle SQL Firewall

You must be granted the appropriate role to administer Oracle SQL Firewall or to query the views that are associated with Oracle SQL Firewall.

To administer Oracle SQL Firewall, you must be granted the `SQL_FIREWALL_ADMIN` role. This role provides the following privileges:

- The `ADMINISTER SQL FIREWALL` system privilege, which is required to run the PL/SQL procedures in the `DBMS_SQL_FIREWALL` package
- The `EXECUTE` privilege for the `DBMS_SQL_FIREWALL` PL/SQL package
- The `READ` privilege for the SQL Firewall `DBA_SQL_FIREWALL_*` data dictionary views

To be able to query the `DBA_SQL_FIREWALL_*` data dictionary views (but not administer SQL Firewall), users must be granted the `SQL_FIREWALL_VIEWER` role.

> **Note:**
>
> The SQL Firewall `SQL_FIREWALL_ADMIN` and `SQL_FIREWALL_VIEWER` roles are powerful roles. Only grant these roles to trusted users.

Related Topics

- Oracle SQL Firewall Data Dictionary Views
  Oracle Database provides a set of data dictionary views that provide information about Oracle SQL Firewall configurations.

1.5 Getting Hands-On Experience with Oracle SQL Firewall

You can use the Oracle LiveLabs workshop for Oracle SQL Firewall to get experience using SQL Firewall.

See the following LiveLabs:

- Get Started with Oracle Data Safe Fundamentals which includes a section on SQL Firewall
- DB Security - SQL Firewall

The following sample demonstration scripts and video of Oracle SQL Firewall in action are also provided for your reference

- Oracle SQL Firewall sample demo scripts

2
Configuring Oracle SQL Firewall

You can configure Oracle SQL Firewall in either an Oracle database using the `DBMS_SQL_FIREWALL` package, or you can configure it in Oracle Data Safe.

- **`DBMS_SQL_FIREWALL` Package:** Use the PL/SQL procedures in the `DBMS_SQL_FIREWALL` package to manage SQL Firewall within an individual Oracle Database instance.
- **Oracle Data Safe:** You can use the Data Safe user interface if you want to manage multiple SQL Firewalls centrally. You can use Data Safe REST APIs, software developer kits (SDKs), CLI, and Terraform for further automation and integration. You can also use the more extensive Oracle Cloud Infrastructure (OCI) ecosystem for integrating SQL Firewall violations with its alerts and notifications.

2.1 Configuring and Managing Oracle SQL Firewall with the DBMS_SQL_FIREWALL Package

After you configure Oracle SQL Firewall for a target user, you can perform maintenance tasks such as modifying the configuration, purging old logs, and troubleshooting errors.

2.1.1 Configuring Oracle SQL Firewall Using the DBMS_SQL_FIREWALL Package

A user who has the `SQL_FIREWALL_ADMIN` role can use the `DBMS_SQL_FIREWALL` PL/SQL package to configure Oracle SQL Firewall in the root or a pluggable database (PDB).

1. Connect to the root or PDB as a user who has been granted the `SQL_FIREWALL_ADMIN` role.
2. Enable SQL Firewall.

   ```
   EXEC DBMS_SQL_FIREWALL.ENABLE;
   ```

3. For every database user to protect with SQL Firewall in the Oracle database, enable SQL Firewall to learn the normal SQL traffic of the database user by capturing all the authorized SQL statements over trusted database connection paths.

   The examples in this procedure assume the user is a PDB user named `APP`. For example:

   ```
   BEGIN
     DBMS_SQL_FIREWALL.CREATE_CAPTURE (
       username         => 'APP',
       top_level_only   => TRUE,
       start_capture    => TRUE
     );
   END;
   /
   ```

   In this specification:

- `username` is the name of the application user that SQL Firewall will monitor. You can only create one capture for each user. You cannot create SQL Firewall captures for the `SYS`, `SYSDG`, `SYSBACKUP`, `SYSRAC`, `SYSKM`, `DVSYS`, `LBACSYS`, or `AUDSYS` users.
- `top_level_only` controls the level of SQL statements that are captured.
  - `TRUE` generates capture logs only for top-level SQL statements, that is, statements that the user directly runs.
  - `FALSE` generates capture logs for both top-level SQL statements and SQL commands issued from PL/SQL units. The default is `FALSE`.
- `start_capture` controls when the capture will be effective.
  - `TRUE` enables SQL Firewall to start capturing the target user's activities right away. The default is `TRUE`.
  - `FALSE` creates a capture for the user, but does not start the capture right away. When you want to start the capture later on, you must run the `DBMS_SQL_FIREWALL.START_CAPTURE` procedure for the user. For example:

    ```
    EXEC DBMS_SQL_FIREWALL.START_CAPTURE ('APP');
    ```

As an application service account, run the normal application SQL workload from the trusted database connection paths when the capture is started for the application service account. In the event of a change in application in the SQL workload following application patching, you may want SQL Firewall to unlearn and learn, starting over. You can delete the current capture, and create a new one. Specifically, if you want to restart the capture process, then you must first stop this capture (if it is started), then either purge the capture logs and start this capture again, or, delete this capture and create (and start) the capture again.

4. Review the capture logs and sessions logs to determine the adequacy of the capture.

   For example:

   ```
   SELECT SQL_TEXT FROM DBA_SQL_FIREWALL_CAPTURE_LOGS WHERE USERNAME = 'APP';
   ```

5. Stop the capture.

   For example:

   ```
   EXEC DBMS_SQL_FIREWALL.STOP_CAPTURE ('APP');
   ```

6. Generate the SQL Firewall policy with allow-lists for the user:

   A SQL Firewall policy with allow-lists sets the baseline for allowed SQL statements and allowed contexts. Allowed SQL statements constitute the approved SQL statements. Allowed contexts represent trusted database connection paths. SQL Firewall creates the allow-list based on data collected from existing capture logs for the user.
   For example:

   ```
   EXEC DBMS_SQL_FIREWALL.GENERATE_ALLOW_LIST ('APP');
   ```

7. To find the permitted and allowed SQL statements that the user can run, query the `DBA_SQL_FIREWALL_ALLOWED_*` data dictionary views.

For example:

```
SELECT SQL_TEXT FROM DBA_SQL_FIREWALL_ALLOWED_SQL WHERE USERNAME = 'APP';
```

To find the trusted database connection paths for the user, perform the following queries:

```
SELECT OS_PROGRAM FROM DBA_SQL_FIREWALL_ALLOWED_OS_PROG WHERE USERNAME =
'APP';


SELECT OS_USER FROM DBA_SQL_FIREWALL_ALLOWED_OS_USER WHERE USERNAME =
'APP';


SELECT IP_ADDRESS FROM DBA_SQL_FIREWALL_ALLOWED_ALLOWED_IP_ADDR WHERE
USERNAME = 'APP';
```

8. Optionally, add or modify entries in the allowed contexts by running the `DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT` and `DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT` procedures.

   You can only add a context after you have generated the allow-list. A context can specify the client IP address, names of operating system users, or the operating system program that can be used for database connections. You can add as many context values as you need. For example, if the user's allowed context list does not contain the IP address 192.0.2.1 but you want to allow the user to connect from this IP after the enablement of the allow-list:

   ```
   BEGIN
     DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT (
       username       => 'APP',
       context_type   => DBMS_SQL_FIREWALL.IP_ADDRESS,
       value          => '192.0.2.1'
      );
   END;
   /
   ```

   To specify all possibilities for a specific `context_type`, enter the `%` wildcard.

   The following three types of `context_type` settings are valid:

   - `DBMS_SQL_FIREWALL.IP_ADDRESS` accepts IPv4 and IPv6 addresses and subnets in the CIDR notation. It accepts the value `Local` (case sensitive) for local connections when the IP address is not available.
   - `DBMS_SQL_FIREWALL.OS_USERNAME` accepts any valid operating system user name, such as `oracle`.
   - `DBMS_SQL_FIREWALL.OS_PROGRAM` accepts any valid operating system program name that the user uses to run SQL statements, such as `sqlplus` or `SQL Developer`.

   You can query the following data dictionary views to check the contexts:

   - `DBA_SQL_FIREWALL_ALLOWED_IP_ADDR`
   - `DBA_SQL_FIREWALL_ALLOWED_OS_USER`
   - `DBA_SQL_FIREWALL_ALLOWED_OS_PROG`

9. Enable the generated SQL Firewall policy to protect the database user.

   The SQL Firewall enforces checks on the allow-lists when the user connects to the database and issues SQL statements.

   This enablement becomes effective immediately, even in the existing sessions of the target user.
   For example:

```
BEGIN
  DBMS_SQL_FIREWALL.ENABLE_ALLOW_LIST (
    username       => 'APP',
    enforce        => DBMS_SQL_FIREWALL.ENFORCE_SQL,
    block          => TRUE
   );
END;
/
```

   In this specification:

   - `username` can be a specific user whose allow-list has been generated, or it can be all users whose allow-list are not currently enabled. To specify all users, use `NULL` as the value.
   - `enforce` specifies one of the following enforcement types:
     - `DBMS_SQL_FIREWALL.ENFORCE_CONTEXT` enforces the allowed contexts that have been configured.
     - `DBMS_SQL_FIREWALL.ENFORCE_SQL` enforces the allowed SQL that has been configured.
     - `DBMS_SQL_FIREWALL.ENFORCE_ALL` enforces both allowed contexts and allowed SQL. This setting is the default.
   - `block` specifies the following:
     - `TRUE` blocks the user's database connection or the user's SQL execution whenever the user violates the allow-list definition.
     - `FALSE` allows unmatched user database connections or SQL commands to proceed. This setting is the default.

     SQL Firewall always generates a violation log for any unmatched user database connection or SQL statement regardless of the enforcement option.

     At this stage, if the user attempts to perform a SQL query that violates the allow-list and you have specified SQL Firewall to block this SQL, then an `ORA-47605: SQL Firewall violation` error appears.

10. Monitor the violation log for abnormal SQL connection attempts or SQL queries that are reported if they are not in allow-list.

    For example:

```
SELECT SQL_TEXT, FIREWALL_ACTION, IP_ADDRESS, CAUSE, OCCURRED_AT
FROM DBA_SQL_FIREWALL_VIOLATIONS WHERE USERNAME = 'APP';
```

Output similar to the following appears:

```
SQL_TEXT                                                    FIREWALL_ACTION
IP_ADDRESS   CAUSE            OCCURRED_AT
------------------------------------------------------------ ---------------
-----------  ---------------- ----------------------------------

SELECT SALARY FROM HR.EMPLOYEES WHERE SALARY >:"SYS_B_0"  BLOCKED
192.0.2.146  Context violation 12-MAY-23 11.12.39.626053 PM +00:00
```

Related Topics

- Configuring and Managing Oracle SQL Firewall with the DBMS_SQL_FIREWALL Package
  After you configure Oracle SQL Firewall for a target user, you can perform maintenance tasks such as modifying the configuration, purging old logs, and troubleshooting errors.
- Oracle SQL Firewall Data Dictionary Views
  Oracle Database provides a set of data dictionary views that provide information about Oracle SQL Firewall configurations.

2.1.2 Modifications to Oracle SQL Firewall Configurations

After you create an Oracle SQL Firewall configuration for a user, you can modify the configuration as necessary.

To find information about Oracle SQL Firewall configurations, you can query the `DBA_SQL_FIREWALL_*` data dictionary views.

Table 2-1 lists operations that you can perform after you have configured SQL Firewall.

Table 2-1 Oracle SQL Firewall Modification Procedures

Operation	Procedure
Enable SQL Firewall	• To enable SQL Firewall in the database, use `DBMS_SQL_FIREWALL.ENABLE`.
Manage captures	• To create a capture, use `DBMS_SQL_FIREWALL.CREATE_CAPTURE`. • To start a capture, use `DBMS_SQL_FIREWALL.START_CAPTURE`. • To modify a capture, delete the current one by using `DBMS_SQL_FIREWALL.DROP_CAPTURE`, and then create a new one by using `DBMS_SQL_FIREWALL.CREATE_CAPTURE`. • To stop the SQL Firewall capture for the specified user, use `DBMS_SQL_FIREWALL.STOP_CAPTURE`. • To delete the SQL Firewall capture for a specified user and delete all the existing capture logs for this user: 1. Use `DBMS_SQL_FIREWALL.STOP_CAPTURE` to stop the capture process. 2. Use `DBMS_SQL_FIREWALL.DROP_CAPTURE` to remove the capture.

Table 2-1 (Cont.) Oracle SQL Firewall Modification Procedures

Operation	Procedure
Manage allow-lists	• To generate an allow-list for a given user, use `DBMS_SQL_FIREWALL.GENERATE_ALLOW_LIST`. • To enable an allow-list for a given user, use `DBMS_SQL_FIREWALL.ENABLE_ALLOW_LIST`. • To update an allow-list enforcement, use `DBMS_SQL_FIREWALL.UPDATE_ALLOW_LIST_ENFORCEMENT`. • To prevent SQL Firewall from capturing and enforcing allow-lists for database connections and SQL executions in Oracle Scheduler jobs, use `DBMS_SQL_FIREWALL.EXCLUDE`. • To append all the SQL from a capture log or violation log (or from both) to the allow-list, use the `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST` procedure. You can run this procedure when the allow-list is either enabled or disabled. The change takes place immediately. • To append a single SQL record from a capture log or violation log to the allow-list, use the `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL` procedure as follows: 1. Query the `DBA_SQL_FIREWALL_VIOLATIONS` or the `DBA_SQL_FIREWALL_CAPTURE_LOGS` data dictionary view to find the target SQL record that you want to add to the allow-list. 2. Enter the obtained `USERNAME`, `SQL_SIGNATURE`, `CURRENT_USER`, and `TOP_LEVEL` values of that record in the `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL` procedure to add the target SQL record to the allow-list. You can run `DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL` when the allow-list is either enabled or disabled. The change takes place immediately. • To export the allow-list of a given user to JSON format into the specified CLOB, use `DBMS_SQL_FIREWALL.EXPORT_ALLOW_LIST`. • To import the allow-list for a given user into a target database, use `DBMS_SQL_FIREWALL.IMPORT_ALLOW_LIST`. • To disable an allow-list for a given user, use `DBMS_SQL_FIREWALL.DISABLE_ALLOW_LIST`. • To add or delete any context values from allowed context lists, use `DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT` or `DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT`, respectively. • To delete any SQL statement from allowed SQL lists, use `DBMS_SQL_FIREWALL.DELETE_ALLOWED_SQL`. • To delete the allow-list for a specified user: 1. Disable the allow-list by using `DBMS_SQL_FIREWALL.DISABLE_ALLOW_LIST`. 2. Use `DBMS_SQL_FIREWALL.DROP_ALLOW_LIST`.
Manage allowed contexts	• To add a specified value to the allowed contexts of a specified user for the given context type, use `DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT`. • To modify an allowed context, delete the current one by using `DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT`, and then create a new one by using `DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT`. • To delete the specified value from the allowed contexts of a specified user for the given context type, use `DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT`.

Table 2-1 (Cont.) Oracle SQL Firewall Modification Procedures

Operation	Procedure
Manage allowed SQL	• To delete the specified entry from the allowed SQL of a specified user, use `DBMS_SQL_FIREWALL.DELETE_ALLOWED_SQL`. You can run this procedure when the allow-list is either enabled or disabled, and the change takes place immediately.
Manage SQL Firewall log tables	• To move the SQL Firewall log tables to a different user-defined tablespace other than the default tablespace, `SYSAUX`: 1. Disable SQL Firewall by using `DBMS_SQL_FIREWALL.DISABLE`. 2. Use the `MOVE` clause of the `ALTER TABLE` statement to perform the move operation. You can also use the `DBMS_SQL_FIREWALL.MOVE_LOG_TABLE` procedure to move the SQL Firewall log tables to another tablespace. • To purge capture logs or violation logs for a user or all users, use `DBMS_SQL_FIREWALL.PURGE_LOG`. • To flush all the SQL Firewall logs that reside in the memory into the log tables, use `DBMS_SQL_FIREWALL.FLUSH_LOGS`.
Disable SQL Firewall	• To disable SQL Firewall in the database and stop all the existing captures and allow-lists that are enabled, use `DBMS_SQL_FIREWALL.DISABLE`.

Related Topics

- *Oracle Database PL/SQL Packages and Types Reference*
- Oracle SQL Firewall Data Dictionary Views
  Oracle Database provides a set of data dictionary views that provide information about Oracle SQL Firewall configurations.

2.1.3 Managing Performance for Capture Logs

Depending on application workloads, Oracle SQL Firewall may generate a large volume of capture logs.

To minimize the adverse impact on database performance, Oracle SQL Firewall relies internally on Fast Ingest for better write performance if sufficient memory is available. To make full use of SQL Firewall, Oracle recommends that you do the following:

- Allocate at least an additional 2G to the `LARGE_POOL_SIZE` parameter setting, on top of the existing `LARGE_POOL_SIZE` requirement.
- Resize the `SGA_TARGET` parameter setting to include this additional requirement. Ensure that the final size is 8G or more.

Related Topics

- *Oracle Database Performance Tuning Guide*

2.1.4 Purging Oracle SQL Firewall Logs

Periodically, you should purge the logs that Oracle SQL Firewall generates by using the `DBMS_SQL_FIREWALL.PURGE_LOG` procedure.

SQL Firewall generates and stores the violation logs in a log table. In an ideal SQL Firewall trained environment, the violation log is not expected to be large. Oracle recommends that you

periodically purge these logs. After you verify that the generated allow-list is valid, you should purge unnecessary logs to reclaim the disk space that the logs are using.

1. Log in to the root or the pluggable database (PDB) where SQL Firewall is configured as a user who has been granted the `SQL_FIREWALL_ADMIN` role.
2. Optionally, as a user who has the `SELECT ANY DICTIONARY` system privilege, query the following data dictionary views to check the logs that you plan to purge:
   - `DBA_SQL_FIREWALL_CAPTURE_LOGS`
   - `DBA_SQL_FIREWALL_VIOLATIONS`
3. Connect to the PDB a user who has been granted the `SQL_FIREWALL_ADMIN` role.
4. Run the `DBMS_SQL_FIREWALL.PURGE_LOG` procedure.

   For example:

```
BEGIN
  DBMS_SQL_FIREWALL.PURGE_LOG (
    username    => 'APP',
    purge_time  => '2023-02-01 00:00:00.00 -08:00',
    log_type    => 'DBMS_SQL_FIREWALL.ALL_LOGS'
   );
 END;
/
```

   In this specification:

   - `username` is the target user for which this SQL Firewall configuration was created. If you omit this value, then Oracle Database purges all logs that match the `purge_time` and `log_type` settings.
   - `purge_time` is the timestamp (in `TIMESTAMP` format) that you can specify to purge only logs that were generated before a certain time. If you omit this value, then Oracle Database purges all logs, regardless of the time when they were generated.
   - `log_type` is the type of the logs to be purged. If you do not specify a value, then the default is `DBMS_SQL_FIREWALL.ALL_LOGS`. Specify one of the following constants:
     - `DBMS_SQL_FIREWALL.CAPTURE_LOG`
     - `DBMS_SQL_FIREWALL.VIOLATION_LOG`
     - `DBMS_SQL_FIREWALL.ALL_LOGS` (default)

Related Topics

- *Oracle Database Reference*

2.1.5 Auditing Oracle SQL Firewall Violations by Using Unified Audit Policies

Oracle recommends that you audit SQL Firewall violations as violations indicate the occurrence of potential abnormal database access patterns.

Auditing SQL Firewall violations with unified auditing records the violation in the database audit trail, `UNIFIED_AUDIT_TRAIL` data dictionary view. It is important that you turn on violation auditing after SQL Firewall is fully trained and the allow-lists of the user is complete, to avoid false positives and reduce unnecessary audit volume.

You can create unified audit policies that are specific to SQL Firewall by specifying the `SQL_FIREWALL` component when you create the unified audit policy. When you query the `UNIFIED_AUDIT_TRAIL`, you can query the `FW_ACTION_NAME` and `FW_RETURN_CODE` columns.

> **Note:**
>
> Oracle Database mandatorily audits all invocations of the SQL Firewall `DBMS_SQL_FIREWALL` PL/SQL administrative procedures.

Related Topics

•

2.1.6 Troubleshooting Oracle SQL Firewall by Enabling or Disabling SQL Firewall Trace Files

As a user who has been granted the `ALTER SESSION` or `ALTER SYSTEM` system privilege, you can generate trace files within the PDB in which you are using Oracle SQL Firewall.

You can set SQL Firewall trace events in both the CDB and in individual PDBs.

- To enable tracing for SQL Firewall, use one of the following statements:

  ```
  ALTER SESSION SET EVENTS 'TRACE[SQL_FIREWALL] DISK=trace_level';
  ALTER SYSTEM SET EVENTS 'TRACE[SQL_FIREWALL] DISK=trace_level';
  ```

  In this specification, replace *`trace_level`* with one of the following values:

  – `LOW` shows the minimum tracing information.

  – `HIGH` shows more detailed tracing information, plus the information returned by `LOW`.

  – `HIGHEST` shows the most detailed tracing information, plus the information returned by `HIGH` and `LOW`.

- To disable tracking for SQL Firewall, use one of the following statements:

  ```
  ALTER SESSION SET EVENTS 'TRACE[SQL_FIREWALL] OFF';
  ALTER SYSTEM SET EVENTS 'TRACE[SQL_FIREWALL] OFF';
  ```

Related Topics

•

2.2 Configuring and Managing Oracle SQL Firewall with Oracle Data Safe

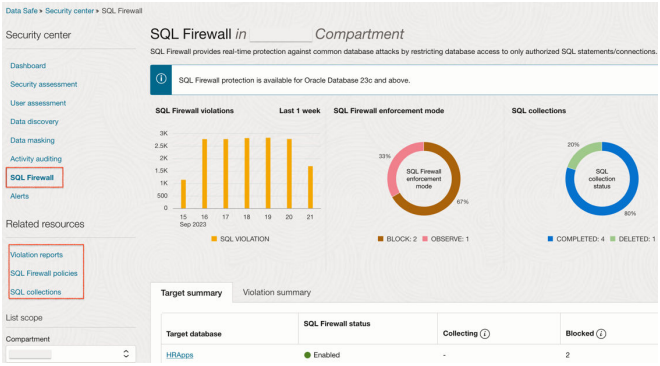
With Oracle Data Safe on Oracle Cloud, you can manage multiple SQL Firewalls centrally and get a comprehensive view of SQL Firewall violations across a fleet of Oracle databases.

SQL Firewall administrators can use Data Safe to collect SQL activities of a database user with its associated database connection paths (IP address, OS program, OS user), and monitor the progress of the collection. Data Safe enables you generate and enable the SQL Firewall policy

from the collected SQL traffic. Data Safe automatically collects the violation logs, and lets you monitor SQL Firewall violations from the console.

The following image shows the SQL Firewall dashboard in Data Safe.

Figure 2-1 SQL Firewall Dashboard in Data Safe



The violation summary in the dashboard provides a comprehensive view of SQL Firewall violations from all the targets in the compartment that have SQL Firewall enabled for the chosen period. From here, you can drill down into the violations for detailed analysis.

Related Topics

- Start Using SQL Firewall
  In order to begin using SQL Firewall you need to complete the following steps. Ensure you have already completed the prerequisites before starting these steps.

2.2.1 SQL Firewall Overview

The SQL Firewall feature in Oracle Data Safe lets you administer and monitor SQL Firewall across your fleet of Oracle Database 23ai targets.

2.2.1.1 About SQL Firewall

The SQL Firewall feature in Oracle Data Safe lets you administer and monitor SQL Firewall across your fleet of Oracle Database targets. SQL Firewall is a new security feature built into the Oracle Database 23ai kernel and offers protection against risks such as SQL injection attacks and compromised accounts. SQL Firewall inspects all incoming database connections and SQL statements, including the ones from PL/SQL units, whether local or over the network, encrypted or clear text. It only allows explicitly authorized SQL and can log or block SQL statements and connections that do not fall within the SQL Firewall allowlists.

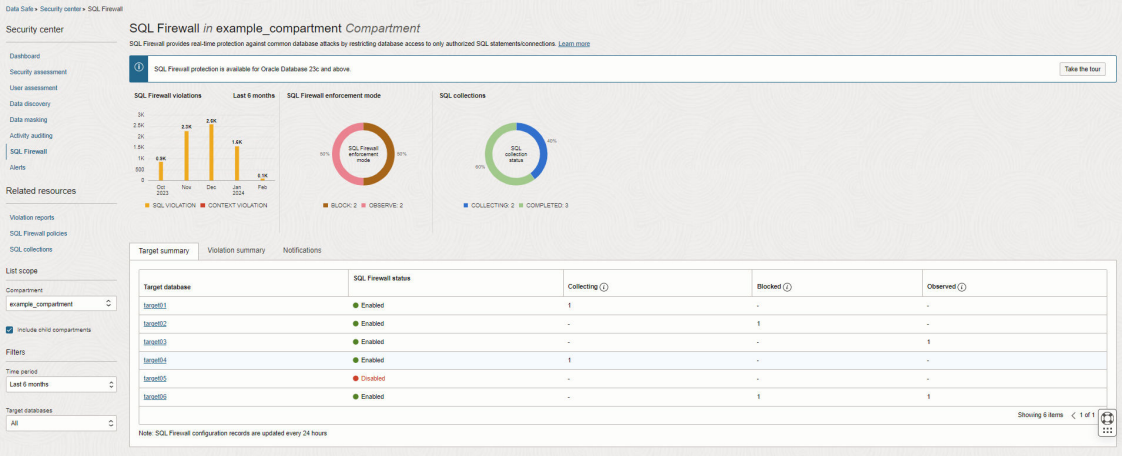
SQL Firewall uses allowlists of authorized SQL statements and trusted database connection paths to determine which SQL statements and connection paths are authorized and which ones should be either logged or blocked. SQL Firewall allowlist policies work at a database account level. You create an SQL Firewall allowlist for a database account by capturing or collecting the expected application SQL workload from expected database connections. Subsequently, the firewall detects and prevents unauthorized SQL and potential SQL injection attacks.

To learn more about SQL Firewall in Oracle Database see the *Oracle Database Oracle SQL Firewall User's Guide*

SQL Firewall can be managed in multiple ways. The PL/SQL procedures in `SYS.DBMS_SQL_FIREWALL` package lets you manage SQL Firewall directly in an Oracle Database (23ai or above). Consider Oracle Data Safe if you are looking forward to leveraging the convenience of the Oracle Cloud Infrastructure (OCI) ecosystem and want to manage and monitor SQL Firewall for a fleet of Oracle Database targets.

Administrators can use Data Safe to collect SQL activities of database accounts, monitor the collection progress, create SQL Firewall policies with allowlist rules (allowed contexts and allowed SQL statements) from the collected SQL activities, and enable SQL Firewall policies. Once a SQL Firewall policy is enabled, Data Safe automatically collects the firewall violation logs from the database and stores them in Data Safe. Those logs are then available for online analysis and reporting across your database fleet as shown in Figure 2-2. You can leverage the Data Safe REST APIs, SDKs, CLI, and Terraform for further automation and integration. You can also leverage the larger OCI ecosystem for integrating SQL Firewall violations with its alerts and notifications.

Figure 2-2 SQL Firewall dashboard in Data Safe



2.2.1.2 Terms in SQL Firewall

The following terms are used throughout Oracle Data Safe's SQL Firewall feature.

- Database security configuration - This resource represents the target database configurations. Included in the Database security configurations are the SQL Firewall configurations such as the status of the firewall, the time that the firewall status was last updated, violation log auto purge settings, and so on.
- Session context - This represents client information initiating SQL traffic: client IP address, OS program name, and OS username.
- SQL collection - This resource represents the SQL collection for a specific database user in a target database. SQL collection encapsulates the SQL commands issued in the user's database sessions and their session context.
- SQL Firewall policy - An allowlist policy specific to a database user through which incoming SQL statements will be evaluated to determine if they can take action on the target database. SQL statements can be allowed or, if they're not part of the allowlist, allowed and logged or blocked and logged. The policy can consist only of session context information, only of specific SQL statements, or both.
- SQL violations - This represents SQL statements that were initiated on the target database that are not included in the allowlist in the SQL Firewall policy.

- Context violations - This represents session context from which SQL statements were initiated on the target database that are not included in the allowlist in the SQL Firewall policy.
- Observe and log violations - A SQL Firewall policy enforcement option where initiated SQL statements that are either SQL or context violations are allowed to execute on the target database and the statements and context are logged for later reference.
- Block and log violations - A SQL Firewall policy enforcement option where initiated SQL statements that are either SQL or context violations are blocked and can't execute on the target database. The statements and context are logged for later reference.

2.2.1.3 Prerequisites for SQL Firewall

SQL Firewall requires you to register an Oracle Database 23ai target database in Data Safe. Users must be granted specific permissions in IAM.

These are the prerequisites for using the SQL Firewall feature in Data Safe:

- Register an Oracle Database 23ai or later. For more information see, Target Registration in the *Administering Oracle Data Safe* guide.
- Grant the SQL Firewall role to the Data Safe service account on the target database. For more information, see Roles for the Oracle Data Safe Service Account in the *Administering Oracle Data Safe* guide.
- Obtain the required IAM permissions which can be granted by a tenancy administrator: To use the full functionality of SQL Firewall it is recommended to be granted `manage` permissions on `data-safe-sql-firewall-family` in the relevant compartments.

```
Allow group <group-name> to manage data-safe-sql-firewall-family in
compartment
        <compartment-name>
```

Alternatively, administrators may grant more selective permissions by granting permissions to specific resources within `data-safe-sql-firewall-family`. For more information on the resources contained within `data-safe-sql-firewall-family`, see `data-safe-sql-firewall-family` Resource.

2.2.2 Start Using SQL Firewall

In order to begin using SQL Firewall you need to complete the following steps. Ensure you have already completed the prerequisites before starting these steps.

These steps will walk you through

1. Enabling SQL Firewall on your Oracle Database 23ai or above
2. Collecting SQL traffic
3. Stopping the collection of SQL traffic
4. Generating and enforcing SQL Firewall policies
5. Viewing SQL Firewall violation logs
6. Creating audit trails and alert policies for SQL Firewall violations
7. Configuring notifications for SQL Firewall violations

By completing these steps you will be taking steps to protect your database fleet against SQL injection attacks and compromised accounts.

2.2.2.1 Step 1: Enable SQL Firewall On Your Target Database

This steps ensures that SQL Firewall is enabled on your target database.

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Click **Enable**. This can be done through either the button under the name of the database security configuration or the Enable option under the Target database section of the Database security configuration information tab.
6. Wait for the resource to change to Active before continuing to the next step.

2.2.2.2 Step 2: Start SQL Collection for a Database User

This step starts the collection of expected SQL statements and expected database connection paths for the database user. Run the typical application workload from the trusted database connection paths.

1. In the Configuration details page from the previous step, click **Create and start SQL collection.**
2. Select the database user for which collection needs to be created.
3. Select the SQL Collection Level:
   - User issued SQL commands - These are SQL statements that were issued directly from the user to be executed on the database. This is the default.
   - User issued SQL commands and SQL commands issues from PL/SQL units - This includes SQL statements issued directly from the user as well as SQL statements within a PL/SQL unit which is invoked by the user.

     Note: SQL collection will *not* record any internal recursive SQL statements.
4. Click **Create and start SQL collection.**
5. Perform typical daily tasks in your applications for the selected database user.
6. Allow the SQL collection to run for some time. This is discussed further in Step 3: Monitor the Progress of SQL Collection with Insights.

2.2.2.3 Step 3: Monitor the Progress of SQL Collection with Insights

In this step you will monitor the collection of SQL statements and determine when collection can be stopped. Monitor the SQL collection until you see there are no new incoming unique SQL statements or trusted connection paths from the running workload.

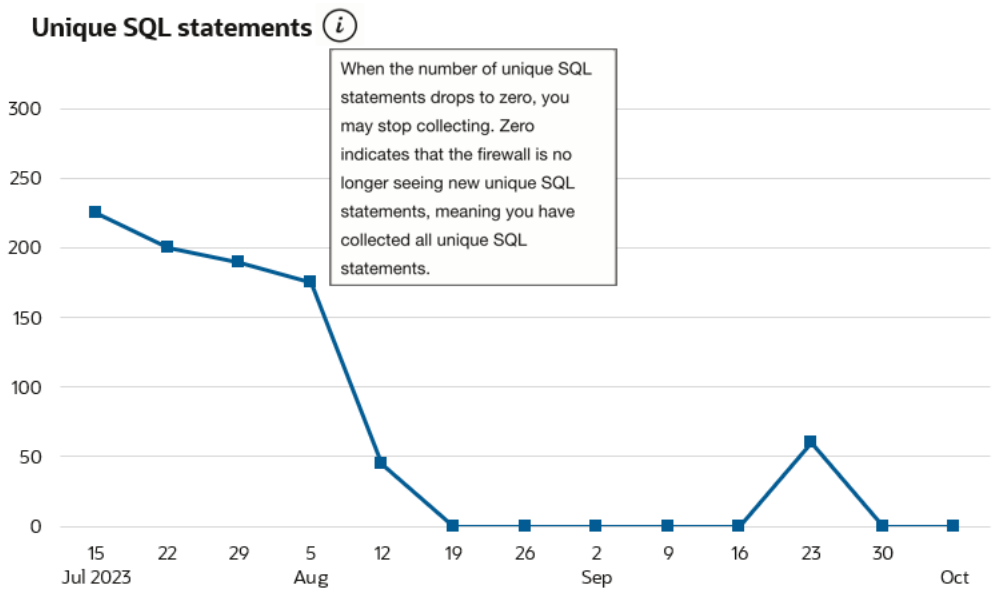
1. Click the **SQL collection insights** tab.

2. The information on the SQL collection tab refreshes every hour, if necessary click **Refresh Insights**.
3. (Optional) Select the time period for which you would like to review the SQL collection.
4. Review the Unique SQL statements chart.
   The collected SQL statements are analyzed to determine if they are unique over the span of the collection period and this chart displays the number of unique SQL statement on the selected time interval. Once there are no more new unique SQL statement being initiated, i.e. the chart remains steady at zero, it is recommended to stop the collection. Waiting until the number of unique SQL statements comes to zero ensures that you collect all statements that are typically executed on your target database and helps establish a status quo.

   For example, if there are 250 SQL statements executed on the first day of the collection but only 225 of those are unique then the chart will show 225 for that day. In the following week if the same 250 statements and an additional 200 new and unique statements are executed then the chart will only show 200. This is because the 250 statements were already collected and observed in week one, thus they are not unique. The number of unique SQL statements will reach zero when there are no more unique SQL statements are observed. See Figure 2-3 for reference.

   It may take several days to weeks for you to collect enough unique SQL statements to stabilize at zero.

Figure 2-3 Unique SQL Statements chart in SQL Collection Insights



5. Review the Client IP, Client OS user, and Client program charts.
   These charts show you the number of client IP addresses, OS users, and programs, respectively, that are executing SQL commands on your target database each day. The specific context information can be viewed in the table below the charts.

   Since SQL statements should be coming from the same session contexts each day, it is recommended to stop the collection when the charts stabilize at a certain value day to day.
6. Review the list of session context types and values.
   Reviewing the list of client IP addresses, client OS users, and client programs allows you to determine where your traffic is coming from. With this information you can set up rules that log or block traffic from all other locations. This is further discussed in Step 4: Generate and Enforce SQL Firewall Policies.

7. Once you have collected a sufficient amount of unique SQL statements, click **Stop** to stop the collection.
   Once you have stopped the collection you will see start time, stop time, and the elapsed time under **Collection timeline** of the **SQL collection information** tab.

2.2.2.4 Step 4: Generate and Enforce SQL Firewall Policies

In this step you will review the information gathered during the collection and create policies with allowlists based on the collected data. Policies will also be enforced to either observe and allow violations or block violations.

1. In the SQL collection details page from the previous step, click **Generate firewall policy.**This will take you to the Firewall policy details page.
2. Review the SQL session context and the Unique allowed SQL statements tables.
   If desired you can add, edit, or remove session context information to be included in the policy but you can't add, edit, or remove any of the collected unique SQL statements in the policy.
3. (Optional) Update the allowed SQL session context values as desired.
   a. Click **Update** for the respective row.
   b. To remove a value, click the **X** at the end of the row in the panel.
   c. To add a value, click **Add** and enter the new value in the empty field.
   d. Click **Update client IP/client program/client OS user**, depending on which context information you selected.
4. (Optional) Download a PDF or XLS report of all Unique allowed SQL statements.
   a. Click **Generate report**.
      A pop-up will appear.
   b. Select which format you want the report in, PDF or XLS.
   c. Enter a name for the report.
   d. Optionally, enter a description for the report.
   e. Click **Generate report**.
   f. Download the report. You have two options:
      - In the Generate report window, click the **here** link. The document will begin downloading.
      - Click **Close** to close the Generate report window. Then, click the **Download report** button. A dialog box is displayed providing you options to open or save the document.
5. Click on **Deploy and Enforce**.
   a. Select the enforcement scope:
      - All (Session contexts and SQL statements)
      - Session contexts only - This option enforces the checks only on the database connection paths.
      - SQL statements only - This option enforces the checks only on the SQL statements.
   b. Select the action on violations:

- Observe (Allow) and log violations - This option will observe and allow all SQL statements and connections to the database while logging any violations.
- Block and log violations - This option will block any SQL statements and database connections not listed in the policy and log the violations. Consider this option when you want SQL Firewall to prevent unauthorized SQL traffic to the database.

c. Audit for violations

- On - This option will write the violation records to the audit trail. It enables alerting and helps demonstrate compliance to your audit requirements. Ensure to start the audit trail in Data Safe to collect the audit events. These audit events contribute to the monthly free limit of 1 million audit records per month per target database.
- Off

d. Click **Deploy and enforce**.

2.2.2.5 Step 5: View SQL Firewall Violation Reports

In this step you can view a report of violations for your enforced SQL Firewall policies. There are a variety of ways to navigate to the violations report, some of which will automatically apply filters for your selected SQL Firewall policies, target databases, time periods, and so on.

> **Note:**
>
> It is unlikely that you will see any violations immediately after enforcing a SQL Firewall policy.

- View all violations
- View target specific violations
- View policy specific violations

View all violations

Complete these steps to view a report of all violations across your database fleet.

1. Under **Security center**, click **SQL Firewall**.
2. Under **Related resources**, click **Violation reports.**
3. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
4. Select which report you would like to see from the **Predefined reports** tab:
   - All violations report - Both SQL and context violations
   - SQL violations report - Violations on SQL statements
   - Context violations report - Violations on database connection paths
5. Once at the report you may perform all standard report actions in Oracle Data Safe such as:
   - Apply basic filters

- Apply advanced SCIM filters
- Create custom reports
- Schedule reports
- Generate and download reports
- Manage which columns to display

View target specific violations

Complete these steps to view a report of all violations on a specific target database from the last week.

1. Under **Security center**, click **SQL Firewall**.
2. Click the **Violation summary** tab.
3. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
4. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
5. Select the name of a target database from the list.
   This will take you to the violation report.
6. The violation report will be automatically filtered to show only the violations for the selected target database from the selected time period.
7. Once at the report you may perform all standard report actions in Oracle Data Safe such as:
   - Apply basic filters
   - Apply advanced SCIM filters
   - Create custom reports
   - Schedule reports
   - Generate and download reports
   - Manage which columns to display

View policy specific violations

Complete these steps to view a report of violations specific to a selected SQL Firewall policy.

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Select the name of a target database from the list on the **Target summary** tab.
   This will take you to the Configuration details page.

5. Under **Resources**, click **SQL Firewall policies**.
6. Select a SQL Firewall policy from the list.
   This will take you to the Firewall policy details page.
7. Under **Enforcement information**, click **View report** next to **Violation reports**.
8. The violation report will be automatically filtered to show only the violations for the selected database user on the selected target database.
   This will take you to the violation report.
9. Once at the report you may perform all standard report actions in Oracle Data Safe such as:
   - Apply basic filters
   - Apply advanced SCIM filters
   - Create custom reports
   - Schedule reports
   - Generate and download reports
   - Manage which columns to display

2.2.2.6 Step 6 (Optional): Create Audit and Alert Policies for SQL Firewall Violations

In this step you will create audit and alerts policies for SQL Firewall violations so that you can better track and monitor activity on your database fleet. Though this step is optional, it is recommended as it enables alerting and helps demonstrate compliance to your audit requirements.

Complete the prerequisites for Activity Auditing and Alerts.

1. Complete the Activity Auditing workflow to audit SQL Firewall violations.
   You need to have turned on **Audit for violations** when enforcing your SQL Firewall policies for the corresponding audit policies to show as enabled in Activity auditing. You can view and manage the audit policies for SQL Firewall listed under the SQL Firewall auditing section of the Audit policy details.

   > **Tip:**
   >
   > You must turn on **Audit for violations** in your SQL Firewall policy before managing the SQL Firewall audit policies in the Activity Auditing workflow. See Update the Enforcement of SQL Firewall Policies for more information.

2. Complete the Alerts workflow to receive alerts for SQL Firewall violations.
   The alert policy for SQL Firewall is **SQL Firewall violations**.

2.2.2.7 Step 7 (Optional): Configure Notifications for SQL Firewall Violations

In this step you will configure notifications for when a SQL Firewall violation occurs. Though this step is optional, it is recommended as it will enable you to receive near real-time alerts in the event of a SQL Firewall violation.

In Data Safe you can create event notifications through a workflow available in SQL Firewall. This allows you to create event notifications in context. You can use the quickstart template for common events or the advanced event notification workflows to create notifications.

Prerequisites:

Ensure you have the necessary IAM permissions to create event notifications. For more information, see Permissions to Use Contextual Event Notifications in the *Administering Oracle Data Safe* guide.

To create notifications:

1. Under **Security center**, click **SQL Firewall**.
2. Click the **Notifications** tab.
3. Click **Create notification**.
   If you don't have any notifications created for the selected resource then you will see a list of available quickstart templates. You may click on one of these instead.

   The **Create notification** side panel will appear.
4. Select to create an event notification from either a **Quickstart** template or an **Advanced event notification**.
   A Quickstart templates allow you to select from a list of common event scenarios. When you create a notification from a quickstart template, the Rule and Event is created automatically.

   > **Note:**
   >
   > The Rule and Event are created in the compartment that you were working in when you started the Notification workflow. Rules and Events will only trigger for the compartment and any child-compartments of the compartment that they were created in.

5. If you selected **Quickstart** in the previous step, make a quickstart **Template selection**.
   If you selected **Advanced event notification** in the previous step, type in a **Rule name** and select an **Event type**.

   See SQL Firewall Event Types in the *Administering Oracle Data Safe* guide for more information on events.
6. Select to either **Create new topic** or to **Select existing topic**.
7. Select a **Compartment**.

   > **Note:**
   >
   > This compartment is where the topic will be created, not where the rule and event will be monitored in.

8. If you're creating a new topic, type the topic name or, if you're using an existing topic, select the topic name.
9. Select a **Subscription protocol**.
10. Provide the necessary inputs for the selected subscription protocol.
11. Optionally, click **Show Advanced Options** to tag the notification.

a. Click **+ Another Tag** to create an additional optional tag to organize and track resources in your tenancy.

b. Select a **Tag Namespace** from the drop-down list.

c. Provide a **Tag Key** and **Tag Value**.

12. Click **Create notification**.

Following the completion of these steps, SQL Firewall will start observing the incoming SQL statements and database connection paths, and will allow or block the SQL traffic to proceed to the target database based on the enforced SQL Firewall policy while logging any violations. You can monitor the SQL Firewall violations in Data Safe. If you configured audit and alert configuration, OCI notifications will be triggered in the event of a SQL Firewall violation.

2.2.3 Gain Insights from SQL Firewall

After successfully setting up SQL Firewall to monitor and block and allow SQL activity on your Oracle Database 23ai target databases, you'll want to ensure that you understand the dashboard and violations report. You should also understand what actions to take in the event of a high volume of violations.

2.2.3.1 View the SQL Firewall Dashboard

When you select **SQL Firewall** under **Security center** in Oracle Data Safe you will see the dashboard of SQL Firewall information for the last week. This dashboard provides you with a high-level view of your SQL Firewall implementation across your fleet of Oracle Database 23ai or above target databases in your selected compartment(s).

To filter the dashboard you can alter the compartments, time period, and databases that you can see information for by:

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.

The dashboard shows the following information:

- **SQL Firewall violations** chart - Shows the number of SQL statement violations and the number of context violations throughout your Oracle Database 23ai or above fleet per day. This allows you to determine patterns in the number of SQL statement and session context violations and identify spikes in violations that should be investigated.
- **SQL Firewall enforcement mode** chart - Shows you a break down of how many of your SQL Firewall policies either "block" or "observe" SQL statements or session contexts that violate your policies.
- **SQL Collections** chart - Shows you a break down of the number of SQL collections in each life cycle state: `COLLECTING`, `COMPLETED`, `DELETED`, `FAILED`, `NEEDS_ATTENTION`.
- **Target Summary** tab - Shows you a break down per registered Oracle Database 23ai or above of the number of database users that SQL statements are actively being collected for, the number of policies that block violations, and the number of polices that allow and observe violations. You can click on the name of a target database to see its SQL Firewall

configuration details and drill down deeper into the SQL collections, SQL Firewall policies, and Work Requests on the target database.

- **Violations Summary** tab - Shows you a break down per registered Oracle Database 23ai or above of the total number of violations, the number of SQL violations, and the number of Context violations. You can click on the name of a target database to see a more detailed violations report.
- The **Notifications** tab - Shows you what event notifications and subscriptions you have created for SQL Firewall. More specifically, it displays the event, rule name, topic name, and when the event notification was created. This table will only show Events that you have created directly within Data Safe. In addition to displaying existing event notifications, you can also create new notifications by using the Create notification button. See Create and Modify Event Notifications in SQL Firewall for more information.

2.2.3.2 View Violations

There are multiple ways that you can view context and SQL statement violations once you have enforced SQL Firewall policies.

- View all violations
- View target specific violations
- View policy specific violations

View all violations

Complete these steps to view a report of all violations across your database fleet.

1. Under **Security center**, click **SQL Firewall**.
2. Under **Related resources**, click **Violation reports.**
3. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
4. Select which report you would like to see from the **Predefined reports** tab:
   - All violations report - Both SQL and context violations
   - SQL violations report - Violations on SQL statements
   - Context violations report - Violations on database connection paths
5. Once at the report you may perform all standard report actions in Oracle Data Safe such as:
   - Apply basic filters
   - Apply advanced SCIM filters
   - Create custom reports
   - Schedule reports
   - Generate and download reports
   - Manage which columns to display

View target specific violations

Complete these steps to view a report of all violations on a specific target database from the last week.

1. Under **Security center**, click **SQL Firewall**.
2. Click the **Violation summary** tab.
3. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
4. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
5. Select the name of a target database from the list.
   This will take you to the violation report.
6. The violation report will be automatically filtered to show only the violations for the selected target database from the selected time period.
7. Once at the report you may perform all standard report actions in Oracle Data Safe such as:
   - Apply basic filters
   - Apply advanced SCIM filters
   - Create custom reports
   - Schedule reports
   - Generate and download reports
   - Manage which columns to display

View policy specific violations

Complete these steps to view a report of violations specific to a selected SQL Firewall policy.

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Select the name of a target database from the list on the **Target summary** tab.
   This will take you to the Configuration details page.
5. Under **Resources**, click **SQL Firewall policies**.
6. Select a SQL Firewall policy from the list.
   This will take you to the Firewall policy details page.
7. Under **Enforcement information**, click **View report** next to **Violation reports**.
8. The violation report will be automatically filtered to show only the violations for the selected database user on the selected target database.
   This will take you to the violation report.

9. Once at the report you may perform all standard report actions in Oracle Data Safe such as:
   - Apply basic filters
   - Apply advanced SCIM filters
   - Create custom reports
   - Schedule reports
   - Generate and download reports
   - Manage which columns to display

Related Topics

- View and Manage Violations Report
  Describes actions that can be take on reports and how to create custom reports.

2.2.3.3 Follow-Up Actions for SQL Firewall

In an ideal scenario where the SQL collection has captured all expected SQL statements and trusted database connections, violations indicate potential database attacks such as compromised account access and SQL Injection attacks. But if the collected statements or database connections are not complete or there are new authorized SQL statements following an application update, there is a possibility to see a surge in violations. Ensure to update the SQL Firewall policies to collect these additional statements to avoid false positives in the violation reports.

Related Topics

- Update SQL Firewall Policies

2.2.4 Manage SQL Firewall

Managing your SQL Firewall policies and configurations helps ensure that your databases are protected from threats while also ensuring that intended SQL actions can be taken on your databases. See the below topics for information on how to update your SQL Firewall configurations and policies.

2.2.4.1 Update the Database Security Configuration

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Perform any of the following tasks:

- Click **Disable** next to **SQL Firewall status** to disable SQL Firewall. This will stop any ongoing collections and policies will no longer be enforced.
- Click **Turn on** or **Turn off** next to **Auto-purge violation logs** to turn this on or off. This specifies whether Data Safe should automatically purge the violation logs from the database after collecting the violation logs and persisting them on Data Safe.

> **Note:**
>
> When this is turned on violation logs are automatically purged every seven days.

- Click **Include** or **Exclude** next to **Database jobs** to include or exclude database jobs for SQL Firewall enforcement.
- Click **Refresh** next to **Last refresh time** to refresh Data Safe's copy of the policies if you made a recent policy change within the database.
- Click **Move Resource** to move the Database Security Configuration to a different compartment.

2.2.4.2 Purge a SQL Collection

Purge helps clean the collection logs for the user. You typically need to purge the SQL Collection when you need to recapture an application SQL workload for the same database user following application updates. The SQL collection can be started again for the database user once it is purged.

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Under **Resources**, click **SQL collections**.
6. Click on a database user name.
   This will take you to the SQL collection details page.
7. Click **Purge** to remove the SQL collection. This will not stop any SQL Firewall Policies that were generated from this collection.

2.2.4.3 Drop a SQL Collection

Drop will remove the SQL Collection and collection logs for the selected database user. You typically have to drop the SQL Collection when you need to remove SQL Firewall protection for a database user who is no longer active or has changed responsibilities in the system.

1. Under **Security center**, click **SQL Firewall**.

2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Under **Resources**, click **SQL Collections**.
6. Click on a database user name.
   This will take you to the SQL collection details page.
7. Click on **More actions** and select **Drop** to delete the SQL collection. Dropping a SQL collection will not have an impact on already generated or enforced SQL Firewall policies.

2.2.4.4 View and Manage SQL Firewall Policies

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Under **Resources**, click **SQL Firewall policies**.
6. Click on a database user name.
   This will take you to the Firewall policy details page.
7. (Optional) Update the allowed SQL session context values as desired.
   a. Click **Update** for the respective row.
   b. To remove a value, click the **X** at the end of the row in the panel.
   c. To add a value, click **Add** and enter the new value in the empty field.
   d. Click **Update client IP/client program/client OS user**, depending on which context information you selected.
8. (Optional) Download a PDF or XLS report of all Unique allowed SQL statements.
   a. Click **Generate report**.
      A pop-up will appear.
   b. Select which format you want the report in, PDF or XLS.
   c. Enter a name for the report.
   d. Optionally, enter a description for the report.
   e. Click **Generate report**.
   f. Download the report. You have two options:

- In the Generate report window, click the **here** link. The document will begin downloading.
- Click **Close** to close the Generate report window. Then, click the **Download report** button. A dialog box is displayed providing you options to open or save the document.

2.2.4.5 Update SQL Firewall Policies

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Under **Resources**, click **SQL collection**.
6. Click on a database user name.
   This will take you to the SQL collections details page.
7. Click on the associated SQL Firewall policy located in the SQL collection information tab.
   This will take you to the Firewall details page.
8. Temporarily disable the SQL Firewall policy by clicking **Disable**. Confirm disablement in the pop-up by clicking **Disable**.
9. Navigate back to the SQL collection by clicking **SQL collection details** in the page breadcrumbs.
10. Click **Start** to capture SQL statements.
11. Initiate the SQL statements you want to add on your target database.
12. Click **Stop** once you have collected the SQL statements.
13. Click **Update firewall policy** to append the new SQL statements to the associated policy.
14. Click on the associated SQL Firewall policy located in the SQL collection information tab.
    This will take you to the Firewall details page.
15. Click on **Deploy and Enforce**.
    a. Select the enforcement scope:
       - All (Session contexts and SQL statements)
       - Session contexts only - This option enforces the checks only on the database connection paths.
       - SQL statements only - This option enforces the checks only on the SQL statements.
    b. Select the action on violations:
       - Observe (Allow) and log violations - This option will observe and allow all SQL statements and connections to the database while logging any violations.

- Block and log violations - This option will block any SQL statements and database connections not listed in the policy and log the violations. Consider this option when you want SQL Firewall to prevent unauthorized SQL traffic to the database.

c. Audit for violations

- On - This option will write the violation records to the audit trail. It enables alerting and helps demonstrate compliance to your audit requirements. Ensure to start the audit trail in Data Safe to collect the audit events. These audit events contribute to the monthly free limit of 1 million audit records per month per target database.
- Off

d. Click **Deploy and enforce**.

2.2.4.6 Update the Enforcement of SQL Firewall Policies

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Under **Resources**, click **SQL Firewall policies.**
6. Select a SQL Firewall policy from the list.
   This will take you to the Firewall policy details page.
7. Click on **Deploy and Enforce**.
   a. Select the enforcement scope:
      - All (Session contexts and SQL statements)
      - Session contexts only - This option enforces the checks only on the database connection paths.
      - SQL statements only - This option enforces the checks only on the SQL statements.
   b. Select the action on violations:
      - Observe (Allow) and log violations - This option will observe and allow all SQL statements and connections to the database while logging any violations.
      - Block and log violations - This option will block any SQL statements and database connections not listed in the policy and log the violations. Consider this option when you want SQL Firewall to prevent unauthorized SQL traffic to the database.
   c. Audit for violations
      - On - This option will write the violation records to the audit trail. It enables alerting and helps demonstrate compliance to your audit requirements. Ensure to start the audit trail in Data Safe to collect the audit events. These audit events contribute to the monthly free limit of 1 million audit records per month per target database.
      - Off

d. Click **Deploy and enforce**.

2.2.4.7 Disable or Enable SQL Firewall Policies

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Under **Resources,** click **SQL Firewall policies.**
6. Select a SQL Firewall policy from the list.
   This will take you to the Firewall policy details page.
7. Click **Disable** or **Enable**. Disabling will stop the SQL Firewall from evaluating any incoming SQL traffic against this SQL Firewall policy. However, this will not delete the policy and it can be enabled again later.

2.2.4.8 Drop SQL Firewall Policies

1. Under **Security center**, click **SQL Firewall**.
2. (Optional) Under **List scope**, select the compartment that contains your target database. Optionally select **Include child compartments** to include target database in the list from child compartments.
3. (Optional) Filter the list of results, under **Filters**, do the following:
   a. (Optional) Select a time period from the **Time period** menu.
   b. (Optional) Select a target database from the **Target database** menu.
4. Click on the name of a target database.
   This will take you to the Configuration details page.
5. Under **Resources**, click **SQL Firewall policies**.
6. Select a SQL Firewall policy from the list.
   This will take you to the Firewall policy details page.
7. Click **Drop**. This will delete the SQL Firewall policy and a SQL Collection will have to be initiated again to re-create this policy.

2.2.5 View and Manage Violations Report

Describes actions that can be take on reports and how to create custom reports.

2.2.5.1 Modifying Columns in a Violations Report

To add or remove columns in the report, do the following:

1. View a predefined or custom violations report.
2. Click on the **Actions** drop down menu.

3. Click **Manage columns**.
   The Manage columns window is displayed.
4. Select columns that you want displayed in the report.
5. Deselect columns that you want to hide in the report.
6. Click **Save changes**.

2.2.5.2 Basic Filtering in a Violations Report

To apply basic filters in the report, do the following:

1. View a custom or predefined violations report.
2. Click **Another filter**.
3. Select a filter type, operator, and enter a value. All columns that are available in the report are available as filter types.
4. Click **Apply**.
5. Repeat steps two through four to apply additional filters.

To remove a filter, click the X beside the filter row.

To filter the report based on a total category (for example, Violations blocked), click the total. The list of violations in the table at the bottom of the report is automatically updated. To remove the filter, click the total again.

> **Note:**
>
> Only some totals in your report are single-click filters.

2.2.5.3 Advanced Filtering in a Violations Report

Advanced filtering of violations can provide flexibility in the way that data is analyzed and reviewed, by allowing organizations to specify complex conditions and multiple criteria that must be met in order for data to be included or excluded from the analysis.

To apply advanced filters in the report, do the following:

1. View a predefined or custom violations report.
2. Click **Show Advanced SCIM Query Builder**.
3. Use the provided filter builder and dropdowns to type in your filter(s). Advanced filtering uses System for Cross-Domain Identity Management (SCIM) syntax and supported operators include:
   - `co`: matches resources with an attribute that contains a given string
   - `eq`: matches resources with an attribute that is equal to a given value (not case sensitive)
   - `eq_cs`: matches resources with an attribute that is equal to a given value (case sensitive)
   - `ew`: matches resources with an attribute that ends with a given string
   - `ge`: matches resources with an attribute that is greater than or equal to a given value
   - `gt`: matches resources with an attribute that is greater than a given value

- `in`: matches resources with an attribute that is equal to any of given values in list
- `le`: matches resources with an attribute that is less than or equal to a given value
- `lt`: matches resources with an attribute that is less than a given value
- `ne`: matches resources with an attribute that is not equal to a given value
- `not_in` : matches resources with an attribute that is not equal to any of given values in list
- `pr`: matches resources with an attribute if it has a given value
- `sw`: matches resources with an attribute that starts with a given string

Operators can be grouped using parentheses to specify the order.

Filters can also be combined using logical operators such as `and` and `or`.

> **Note:**
>
> If you have any basic filters currently applied they will appear in the query builder as well.

4. Click **Apply**.

To clear the query builder, click **Clear**. This will clear any basic filters applied as well.

Example 2-1 Context violations and SQL violations that are allowed advanced filter

```
(violationAction eq "ALLOWED")  and ((violationCause eq "context violation")
or (violationCause  eq "SQL violation"))
```

Example 2-2 SQL violations on a specific target database advanced filter

```
(targetName eq "HRApps") and (violationCause eq "SQL violation")
```

Example 2-3 Actions taken on two specific databases since a specifc time advanced filter

```
(operationTime ge "2023-09-11T00:39:43.295Z") and ((targetName eq "HRApps")
or (targetName eq "TF_AUTOMATION"))
```

2.2.5.4 Tips for Using the Filter Builder to Create Advanced Filters

- Pressing the escape key while in advanced filtering mode will clear the whole query.
- Pressing the space key will display the drop down with the list of available attributes or operators.
- Pressing the space key after entering a value like `targetname (demo_tgt)` will enclose the string with quotes: `("demo_tgt")`.
- Pressing enter will close the drop down listing the operators and attribute names.
- If a value like SQL Firewall policy name has spaces in it, typing space will enclose the first word within quotes, `"policy name"`. You will have to move the cursor back to the enclosed string and continue typing the rest of the string value.

- If you build a filter in advanced filtering that can't be displayed in basic filters, you can't switch back to basic filtering mode. For example, advanced filters with the or condition can't be displayed in basic filtering.
- A custom report with basic filter can be updated with advanced filter and saved.

For more information about SCIM, see the protocol documentation at https://www.rfceditor.org/rfc/rfc7644.

For more information about filtering in SCIM, see the filtering section of the protocol documentation at https://www.rfc-editor.org/rfc/rfc7644#section-3.4.2.2.

2.2.5.5 Create a Custom Violations Report

You can create a custom report from any violations report, including the predefined AllViolations report. The details saved to the custom reports are those that you are currentlyviewing on screen. You may want to create a custom report if you want to preserve thefilters and columns displayed in a report that you are viewing online. You may alsowant to store your custom reports in specific compartments.

1. Under **Security center**, click **SQL Firewall**.
2. Under **Related resources**, click **Violation reports**.
3. Click a report name and modify it as needed. If there aren't any custom reports saved, click the All violations report and make changes to it.
4. Click **Create custom report**.
   The Create custom report dialog box is displayed.
5. Enter a name for your custom report.
6. (Optional) Enter a description for your custom report.
7. Select the compartment to where you want to save your custom report.
8. Click **Create custom report**, and wait for a message that tells you the custom report is created.
9. (Optional) To open and view your custom report, click the **click here** link.
10. (Optional) To return to the report displayed on the screen, click **Close**.

2.2.5.6 Update a Custom Violations Report

1. Under **Security center**, click **SQL Firewall**.
2. Under **Related resources**, click **Violation reports**.
3. Click **Custom reports** tab.
4. Click a custom report name.
5. Modify the report as needed.
6. Click **Save Report**.
   The custom report is updated.

2.2.5.7 Delete a Custom Violations Report

When you delete a custom violations report, the report is permanently deleted and cannot berecovered. You cannot delete the predefined All violations report.

1. Under **Security center**, click **SQL Firewall**.

2. Under **Related resources**, click **Violation reports**.
3. Click **Custom reports** tab.
4. Click a custom report name.
5. Click **Delete report**.

   A Delete report dialog box is displayed, asking you to confirm the deletion.
6. Click **Delete report**.

2.2.5.8 Create or Manage a Schedule for a Violations Report

You can create a schedule for a predefined or custom violation report to generate a PDF or XLS report.

1. Under **Security center**, click **SQL Firewall**.
2. Under **Related resource**, click **Violation reports**.
   The **Violation reports** page is displayed, showing you a list of violation reports.
3. To view a predefined violation report, on the **Predefined reports** tab, in the **Report name** column, click the report name that you want to view.
   The predefined report is displayed.
4. To view a custom violation report, click the **Custom report** tab. In the **Report name** column, click the name of your custom report.
   Your custom report is displayed.
5. Click **Manage report schedule**.
   The **Manage report schedule** panel is displayed, pre-loaded with either the default or modified schedule.
6. (Optional) In the **Schedule report name** box, enter a name for the PDF or XLS report.
7. Select a compartment to store the reports generated by the schedule.
8. For **Report format**, select either a **PDF** or **XLS** output.
9. Select a **Schedule frequency**.
   - If you select weekly, select the day of the week in the **Every** field.
   - If you select monthly, select the day of the month in the **Day** field.
10. In **Time (in UTC)**, select a schedule time.
11. In **Events time span**, select the time span for the violation records.
    For example, selecting Last months and entering 14 pulls violations from the last 14 months from the time the report is run.
12. (Optional) Specify a row limit. If unspecified, the default row limit is 200 rows.
13. Click **Save Schedule**.
    You can access the generated PDF/XLS reports on the **Violation report history** page.

2.2.5.9 View and Manage Violation Report History

The **Violation report history** page lists all the PDF/XLS violations reports that are automatically generated via a schedule or on-demand by users. On this page, you can view the list of reports generated during the past three months, details about those reports, and download reports. Oracle Data Safe stores these reports for up to three months.

1. Under **Security center**, click **SQL Firewall**.

2. Under **Related resources**, click **Violation report history**.
   The Violation report history table is displayed. It contains the following information:
   - **Name** - The name of the violation report
   - **State** - Either **Active** or **Updating**, shows if the report is currently accessible or if it is being updated
   - **Report definition** - Specifies the name of the report that provides data for this scheduled or generated report
   - **Generated time** - The time the report was created
   - **Report type** - Generated or Scheduled. Where generated reports are on-demand reports produced outside of the scheduling system and scheduled reports are those produced by the scheduling system
   - **File format** - PDF or XLS
   - **Download report** - Option to download the report
3. (Optional) Under **Filters**, narrow down the report history page based on the **Report definition**, **Report type**, **File format**, and **Time period**.
4. Click on any report name to see further details including OCID and compartment information.

2.2.6 Create and Modify Event Notifications in SQL Firewall

You can create and modify event notifications in SQL Firewall.

2.2.6.1 Creating Event Notifications for SQL Firewall

In Data Safe you can create event notifications for SQL Firewall related events. You can use the quickstart template for common events or the advanced event notification workflows to create notifications.

Prerequisites:

Ensure you have the necessary IAM permissions to create event notifications. For more information, see Permissions to Use Contextual Event Notifications in the *Administering Oracle Data Safe* guide.

To create notifications:

1. Under **Security center**, click **SQL Firewall**.
2. Click the **Notifications** tab.
3. Click **Create notification**.
   If you don't have any notifications created for the selected resource then you will see a list of available quickstart templates. You may click on one of these instead.

   The **Create notification** side panel will appear.
4. Select to create an event notification from either a **Quickstart** template or an **Advanced event notification**.
   A Quickstart templates allow you to select from a list of common event scenarios. When you create a notification from a quickstart template, the Rule and Event is created automatically.

> **Note:**
>
> The Rule and Event are created in the compartment that you were working in when you started the Notification workflow. Rules and Events will only trigger for the compartment and any child-compartments of the compartment that they were created in.

5. If you selected **Quickstart** in the previous step, make a quickstart **Template selection**. If you selected **Advanced event notification** in the previous step, type in a **Rule name** and select an **Event type**.

   See SQL Firewall Event Types in the *Administering Oracle Data Safe* guide for more information on events.

6. Select to either **Create new topic** or to **Select existing topic**.
7. Select a **Compartment**.

> **Note:**
>
> This compartment is where the topic will be created, not where the rule and event will be monitored in.

8. If you're creating a new topic, type the topic name or, if you're using an existing topic, select the topic name.
9. Select a **Subscription protocol**.
10. Provide the necessary inputs for the selected subscription protocol.
11. Optionally, click **Show Advanced Options** to tag the notification.
    a. Click **+ Another Tag** to create an additional optional tag to organize and track resources in your tenancy.
    b. Select a **Tag Namespace** from the drop-down list.
    c. Provide a **Tag Key** and **Tag Value**.
12. Click **Create notification**.

2.2.6.2 Modifying Event Notifications For SQL Firewall

After creating event notifications in SQL Firewall in Oracle Data Safe, you can modify the notifications you created.

To modify the event and rule:

1. Under **Security center**, click **SQL Firewall**.
2. Click the **Notifications** tab.
3. Click on an existing event from the **Name** column.

> **Note:**
>
> You will only see the Events that were created directly within Data Safe.

This will bring you to the Rule details page which is part of Oracle Cloud Infrastructure (OCI) Events Service. For more information, see the Events section of the OCI Documentation.

To modify the topic and subscription:

1. Under **Security center**, click **SQL Firewall**.
2. Click the **Notifications** tab.
3. Click on an existing topic from the **Topic** column.

> **Note:**
>
> You will only see the Topics that were created directly within Data Safe.

This will bring you to the Topic Details page which is part of Oracle Cloud Infrastructure (OCI) Notifications. For more information, see the Notifications section of the OCI Documentation.

3

How Oracle SQL Firewall Works with Other Oracle Features

Learn how Oracle SQL Firewall works in conjunction with other Oracle features.

3.1 SQL Firewall and Audit Vault and Database Firewall (AVDF)

AVDF 20.13 and later can collect SQL Firewall violation logs.

Learn how to configure AVDF to collect SQL Firewall violation logs: Using SQL Firewall with AVDF.

3.2 Oracle SQL Firewall and Oracle Data Pump

You can use Oracle Data Pump to export and import Oracle SQL Firewall captures and allow-list metadata.

3.2.1 About Oracle Data Pump Export and Import Operations on Oracle SQL Firewall Metadata

Oracle SQL Firewall integrates with Oracle Data Pump to support the export and import of the SQL Firewall metadata, including the metadata for captures and allow-lists.

This is typically required in scenarios where the training can be done once on a non-production database, and then SQL Firewall can be enabled on multiple production databases using the allow-list that was generated during the non-production training stage.

Oracle Database maintains the status of captures and allow-lists during the export and import operations, unless you are merging an allow-list from the source database into an existing allow-list in the target database. For example, if a capture is enabled in the source database at the export time, it will be enabled in the target database after the import operation completes. This is similar if you are importing an allow-list when there is no allow-list for the same user in the target database before the import operation.

If you are merging an allow-list from the source database into an existing allow-list in the target database, the settings (such as `status`, `top_level_only`, `enforce`, and `block`) of the allow-list in the target database remain the same as before the import operation. Only the allowed SQL and contexts are merged.

For Oracle Data Pump, Oracle supports the export or import of all the existing SQL Firewall metadata (that is, captures and allow-lists) as a whole. Oracle does not support the export or import of a specific capture or a specific allow-list through Oracle Data Pump.

If you only want to export or import the allow-list for one user, from one specific database to another, then use the `DBMS_SQL_FIREWALL.EXPORT_ALLOW_LIST` or `DBMS_SQL_FIREWALL.IMPORT_ALLOW_LIST` procedure. (These two procedures do not rely on Oracle Data Pump and can be used independently.) Oracle does not support the export and import of SQL Firewall logs (that is, capture and violation logs).

3.2.2 Cases Where Oracle Data Pump Skips the Import for an Oracle SQL Firewall Capture or Allow-List

During an import operation, Oracle Data Pump will skip a particular Oracle SQL Firewall capture or allow-list and continue to import other captures or allow-lists for certain cases.

These cases are as follows:

- If the target users do not exist in the target database, then the captures and allow-lists for those non-existing users are not imported.
- If an allow-list refers to one or more current users that do not exist in the target database, then this allow-list is not imported.
- For an allow-list to be imported, if an allow-list for the same user already exists in the target database and its `top_level_only` setting is different than the allow-list to be imported, then the allow-list is not imported.
- For an allow-list to be imported, if a capture for the same user already exists in the target database and its `top_level_only` setting is different than the allow-list to be imported, then the allow-list is not imported.
- If an allow-list to be imported is enabled, and in the target database, there is an enabled capture for the same user but there is no disabled allow-list for the same user, then the allow-list is not imported to avoid having an enabled capture and an enabled allow-list for the same user at the same time.
- If a capture to be imported already exists for the same user in the target database, then the capture is not imported.
- If a capture to be imported is enabled, and there is an enabled allow-list for the same user in the target database, then the capture is not imported to avoid having an enabled capture and an enabled allow-list for the same user at the same time.
- For a capture to be imported, if an allow-list for the same user already exists in the target database and its `top_level_only` setting is different than the capture to be imported, then the capture is not imported.

3.2.3 Using Oracle Data Pump with Oracle SQL Firewall

You can use the `expdp` and `impdp` commands to export and import Oracle SQL Firewall captures and allow-lists metadata.

1. Log in to the server where SQL Firewall is used.
2. At the command line, perform the Oracle Data Pump export or import operation.
   - To export SQL Firewall metadata, use the following syntax:

     ```
     expdp user_name@pdb_name FULL=Y DIRECTORY=dumpfile_dir
     INCLUDE=SQL_FIREWALL dumpfile=dumpfile_name.dmp LOGFILE=filename.log
     ```

     In this specification:

     – `FULL=Y`, which enables full export mode. SQL Firewall metadata will be exported only with the full export mode.

  - `INCLUDE=SQL_FIREWALL` can be used in the `INCLUDE` or `EXCLUDE` filter. This tag is optional. It enables you to export and import just the SQL Firewall metadata from one database to another.

  For example:

```
expdp "hr@hr_pdb" FULL=Y DIRECTORY=sql_fw_dumpfiles
INCLUDE=SQL_FIREWALL DUMPFILE=sql_fw_app.dmp LOGFILE=sql_fw_app.log
Enter password: password
```

- To import SQL Firewall metadata:

```
impdp user_name@pdb_name FULL=Y DIRECTORY=dumpfile_dir
INCLUDE=SQL_FIREWALL dumpfile=dumpfile_name.dmp LOGFILE=filename.log
```

  For example:

```
impdp "hr@hr_pdb" FULL=Y DIRECTORY=dumpfile_dir INCLUDE=SQL_FIREWALL
dumpfile=sql_fw_app.dmp LOGFILE=sql_fw_app.log
Enter password: password
```

Related Topics

- *Oracle Database Utilities*

3.3 Oracle SQL Firewall and Oracle Scheduler Jobs

In most scenarios, you may want to exclude Oracle Scheduler jobs from Oracle SQL Firewall enforcement because these are not typically run by users.

By default the Oracle Scheduler jobs are excluded. You can enable or disable the enforcement of SQL Firewall during Oracle Scheduler operations by setting the `FEATURE` parameter to the `DBMS_SQL_FIREWALL.SCHEDULER_JOB` constant, using the following procedures:

- `DBMS_SQL_FIREWALL.INCLUDE` permits SQL Firewall to capture any SQL or enforce any allow-lists during Oracle Scheduler operations.
- `DBMS_SQL_FIREWALL.EXCLUDE` prevents SQL Firewall from capturing any SQL or enforcing any allow-lists during Oracle Scheduler operations.

For example:

```
EXEC DBMS_SQL_FIREWALL.EXCLUDE (DBMS_SQL_FIREWALL.SCHEDULER_JOB);
```

Related Topics

- *Oracle Database PL/SQL Packages and Types Reference*
- Oracle SQL Firewall Data Dictionary Views
  Oracle Database provides a set of data dictionary views that provide information about Oracle SQL Firewall configurations.

3.4 Oracle SQL Firewall and Oracle Database Vault

Oracle Database Vault requires special authorization before you can use Oracle SQL Firewall in a Database Vault environment.

3.4.1 Using SQL Firewall in an Oracle Database Vault Environment

Depending on the type of protection that you want to configure, you can use either or both Oracle Database Vault and SQL Firewall.

Database Vault enables you to use realms and command rules to block access to sensitive objects, the execution of critical commands, and SQL connections from untrusted factors such as the time of the day, IP address, host name, program name, or any number of identifiable attributes that are associated with the user. In a Database Vault environment, you can extend this protection by using SQL Firewall to capture an allow-list of SQL commands with an associated trusted database connection paths for a database account. Then you can log (and optionally block) the unseen SQL traffic. SQL Firewall enforcement can distinguish approved SQL statements and connections from the unauthorized SQL traffic, which adds to the protection layer that realms and command rules provide to prevent access to sensitive objects unless they have been explicitly authorized.

The following table shows a comparison of how you can enforce protections using Database Vault realms and command rules, and SQL Firewall.

Table 3-1 Comparison of Oracle Database Vault and SQL Firewall Protections

Use Case	Realms	Command Rules	SQL Firewall
Protect database schemas	Yes, traditional or mandatory realms can limit access to your data. • Entire schema or schemas • Object types • Specific objects by name	Yes, DML or DDL statements against schema objects	No
Protect database roles	Yes, traditional or mandatory realms can protect your roles.	Yes, create a command rule with `GRANT` or `REVOKE` statements for specific roles.	No
Protect database objects	Yes, traditional or mandatory realms can limit access to your data. • Entire shema or schemas • Object types • Specific objects by name	Yes, DML or DDL statements against schema objects • Entire schema or schemas • Object types • Specific objects by name	No
Protect individual SQL statements	No	Yes, control statements against schema or individual schema objects.	Yes, block all but explicitly allowed SQL statements.
Allow-list and protect application SQL traffic	No	No	Yes, block all but explicitly allowed SQL statements.
Protect against risks of compromised accounts	Yes, establish trusted path conditions based on any factors that can be checked programmatically.	Yes, protect `CONNECT` command usage.	Yes, block sessions from untrusted client IP, program and OS user name
Protect database users against SQL Injection risks	No	No	Yes, create an allow-list SQL Firewall policy for each database user and enforce it.

3.4.2 Authorization for Using SQL Firewall in an Oracle Database Vault Environment

In an Oracle Database Vault environment, users who want to configure SQL Firewall must have Oracle Database Vault-specific authorization.

When Database Vault is enabled, the management of SQL Firewall (that is, the invocation of the `DBMS_SQL_FIREWALL` package) requires SQL Firewall administrators to have Database Vault-specific authorization in addition to the `ADMINISTER SQL FIREWALL` system privilege. This requirement is to ensure that only trusted users will be able to manage SQL Firewall in a Database Vault environment.

You can authorize SQL Firewall administrators to allow or not allow captures on users who have the `DV_OWNER`, `DV_ADMIN`, or `DV_ACCTMGR` roles in a Database Vault environment. When Database Vault operations control is enabled, common users will be blocked from using SQL Firewall (that is, the `DBMS_SQL_FIREWALL` procedures for managing captures and allow-lists) on local users unless the common users are included in the exception list.

Related Topics

- *Oracle Database Vault Administrator's Guide*

3.5 Oracle SQL Firewall and Oracle Real Application Security

You can use Oracle SQL Firewall with Oracle Real Application Security (Oracle RAS) to capture SQL statements that come from an Oracle RAS application for the `XS$NULL` user.

You can generate and enforce an allow-list for the `XS$NULL` user after completing a SQL Firewall capture operation. However, SQL Firewall does not perform capture and enforce operations for Oracle RAS end-user identities.

3.6 Oracle SQL Firewall and Oracle Database Centrally Managed Users and Enterprise Users

Oracle SQL Firewall will capture a global user's activities if the SQL Firewall capture is enabled.

However, SQL Firewall does not distinguish enterprise user identities (for example, centrally managed users with Active Directory (CMU-AD) users, Oracle Cloud Infrastructure (OCI) Identity and Access Management (IAM) users, Oracle Internet Directory (OID) users, or Microsoft Azure Active Directory users).

3.7 Oracle SQL Firewall and Oracle Virtual Private Database

When Oracle Virtual Private Database policies are run, Oracle SQL Firewall captures SQL commands right after their executions.

However, SQL Firewall does not consider any modification or transformation that is made by the database kernel (for example, views, synonyms, SQL macro expansion, Virtual Private Database enforcement, and so on). You should train SQL Firewall to capture all the expected incoming SQL statements to formulate the allow-list.

3.8 Oracle SQL Firewall in a Multitenant Environment

Oracle SQL Firewall is affected at both the CDB root level and the individual PDB level.

You can run the SQL Firewall processes and set SQL Firewall trace events in both the CDB and individual PDBs.

In the CDB root:

- You can enable SQL Firewall in the CDB root container, and then create SQL Firewall policies, enable or disable SQL Firewall, start or stop captures, and enable or disable allow-lists. These settings apply to the CDB root only.
- In an Oracle Database Vault operations control environment, there are no restrictions in using SQL Firewall.

In individual PDBs:

- You can enable SQL Firewall in an individual PDB, and then create SQL Firewall policies, enable or disable SQL Firewall, start or stop captures, and enable or disable allow-lists. These settings apply to the current PDB only.
- In a Database Vault operations control environment, common users cannot start or stop captures on local users, nor can they enable or disable allow-lists on local users.

4

Oracle SQL Firewall Data Dictionary Views and Example Queries

Oracle provides a set of data dictionary views that enable you to find different kinds of information about the Oracle SQL Firewall protections that you have configured.

4.1 Oracle SQL Firewall Data Dictionary Views

Oracle Database provides a set of data dictionary views that provide information about Oracle SQL Firewall configurations.

Table 4-1 lists these data dictionary views.

Table 4-1 Data Dictionary Views That Display Oracle SQL Firewall Information

View	Description
DBA_SQL_FIREWALL_ALLOW_LISTS	Lists the status and generation date of the user's allow-lists
DBA_SQL_FIREWALL_ALLOWED_IP_ADDR	Lists the allowed IP addresses for a user
DBA_SQL_FIREWALL_ALLOWED_OS_PROG	Lists the allowed operating system programs for a user
DBA_SQL_FIREWALL_ALLOWED_OS_USER	Lists the allowed operating system users for a user
DBA_SQL_FIREWALL_ALLOWED_SQL	Lists information about the allowed SQL statements for a user, including the allowed SQL ID and the allow-list version of the allowed SQL
DBA_SQL_FIREWALL_CAPTURE_LOGS	Lists log information for a user's SQL Firewall configuration, such as the database user name, SQL text, accessed objects, and the SQL Firewall session ID
DBA_SQL_FIREWALL_CAPTURES	Lists the status SQL Firewall captures, such as whether they are enabled
DBA_SQL_FIREWALL_SESSION_LOGS	Lists information about the SQL Firewall session, such as the session ID, database user name, and client program
DBA_SQL_FIREWALL_SQL_LOGS	Lists information about the SQL logs, such as the SQL text, the command type, the SQL signature, accessed objects, and the character set
DBA_SQL_FIREWALL_STATUS	Lists the status of an SQL Firewall configuration, such as whether it is enabled and what its timestamp is
DBA_SQL_FIREWALL_VIOLATIONS	Provides a detailed report on SQL Firewall violations, including information such as the objects that were accessed, the user the SQL was run on, and whether the action was blocked or allowed

Related Topics

- *Oracle Database Reference*

4.2 Query to Find a User's Allowed SQL and Accessed Objects

The `DBA_SQL_FIREWALL_ALLOWED_SQL` data dictionary view shows the SQL that a user is allowed to use.

For example:

```
SELECT SQL_TEXT, ACCESSED_OBJECTS FROM DBA_SQL_FIREWALL_ALLOWED_SQL WHERE USERNAME =
'HR';

SQL_TEXT                          ACCESSED_OBJECTS
--------------------------------- ------------------
SELECT COUNT(*) FROM HR.EMPLOYEES   "HR"."EMPLOYEES'
```

Related Topics

- *Oracle Database Reference*

4.3 Query to Find a User's Allowed IP Address

The `DBA_SQL_FIREWALL_ALLOWED_IP_ADDR` data dictionary view shows the IP address that a user is allowed to use.

For example:

```
SELECT IP_ADDRESS FROM DBA_SQL_FIREWALL_ALLOWED_IP_ADDR WHERE USERNAME = 'HR';

IP_ADDRESS
------------
192.0.2.1
```

Related Topics

- *Oracle Database Reference*

4.4 Query to Find a User's Oracle SQL Firewall Violations

The `DBA_SQL_FIREWALL_VIOLATIONS` data dictionary view shows the Oracle SQL Firewall violations that a user has committed.

For example:

```
SELECT SQL_TEXT, OCCURRED_AT, FIREWALL_ACTION FROM DBA_SQL_FIREWALL_VIOLATIONS WHERE
USERNAME = 'HR';

SQL_TEXT                          OCCURRED_AT                          FIREWALL_ACTION
--------------------------------- ----------------------------------   ---------------
SELECT COUNT(*) FROM HR.EMPLOYEES  12-OCT-23 10.30.02.238383 AM +00:00 BLOCKED
```

Related Topics

- *Oracle Database Reference*

Appendix: SQL Firewall Database Views and DBMS Package

See the SQL Firewall Database Views and the SQL Firewall DBMS package.

A

SQL Firewall Database Views

DBA_SQL_FIREWALL Parameters

A.1 DBA_SQL_FIREWALL_ALLOW_LISTS

DBA_SQL_FIREWALL_ALLOW_LISTS displays information about SQL Firewall allow-lists.

Column	Datatype	NULL	Description
USERNAME	VARCHAR2(128)		Name of the target user
GENERATED_ON	TIMESTAMP(6) WITH TIME ZONE	NOT NULL	Date and time of allow-list generation
STATUS	VARCHAR2(8)		Allow-list status (ENABLED or DISABLED)
STATUS_UPDATED_ON	TIMESTAMP(6) WITH TIME ZONE	NOT NULL	Date and time of the most recent allow-list status change
TOP_LEVEL_ONLY	VARCHAR2(14)		Indicates whether the allow-list should be enforced on only top-level SQL commands, that is, SQL commands issued directly from the user (Y) or on all SQL commands, including top-level SQL commands and SQL commands from PL/SQL units (N)
ENFORCE	VARCHAR2(15)		Option of the allow-list enforcement: • ENFORCE_CONTEXT - Allowed contexts will be checked and enforced during database connection • ENFORCE_SQL - Allowed SQLs will be checked and enforced for every SQL statement execution • ENFORCE_ALL - Both allowed contexts and allowed SQLs will be checked and enforced
BLOCK	VARCHAR2(14)		If the allow-list is enabled, indicates whether the allow-list is enabled in blocking mode (Y) or non-blocking mode (N)

> **Note:**
>
> This view is available starting with Oracle Database 23ai.

A.2 DBA_SQL_FIREWALL_ALLOWED_IP_ADDR

DBA_SQL_FIREWALL_ALLOWED_IP_ADDR lists the IP addresses that SQL Firewall target users are allowed to use to connect to the database.

Column	Datatype	NULL	Description
USERNAME	VARCHAR2(128)		Name of the target user
IP_ADDRESS	VARCHAR2(128)	NOT NULL	Allowed client IP address

> **Note:**
>
> This view is available starting with Oracle Database 23ai.

A.3 DBA_SQL_FIREWALL_ALLOWED_OS_PROG

DBA_SQL_FIREWALL_ALLOWED_OS_PROG lists the OS programs that SQL Firewall target users are allowed to use to connect to the database.

Column	Datatype	NULL	Description
USERNAME	VARCHAR2(128)		Name of the target user
OS_PROGRAM	VARCHAR2(128)	NOT NULL	Allowed OS program name

> **Note:**
>
> This view is available starting with Oracle Database 23ai.

A.4 DBA_SQL_FIREWALL_ALLOWED_OS_USER

DBA_SQL_FIREWALL_ALLOWED_OS_USER lists the OS user names that SQL Firewall target users are allowed to use to connect to the database.

Column	Datatype	NULL	Description
USERNAME	VARCHAR2(128)		Name of the target user
OS_USER	VARCHAR2(128)	NOT NULL	Allowed OS user name

> **Note:**
>
> This view is available starting with Oracle Database 23ai.

A.5 DBA_SQL_FIREWALL_ALLOWED_SQL

DBA_SQL_FIREWALL_ALLOWED_SQL displays information about allowed SQL for SQL Firewall allow-lists.

Column	Datatype	NULL	Description
USERNAME	VARCHAR2(128)		Name of the target user
ALLOWED_SQL_ID	NUMBER	NOT NULL	Unique ID of allowed SQL entry for the target user
SQL_SIGNATURE	VARCHAR2(64)	NOT NULL	Allowed SQL signature
SQL_TEXT	VARCHAR2(4000)		SQL text (up to 1000 characters) To view the full SQL text, query the SQL_TEXT column of the DBA_SQL_FIREWALL_SQL_LOGS view.
ACCESSED_OBJECTS	VARCHAR2(4000)		List of accessed objects (up to 1000 characters) To view the full list of accessed objects, query the ACCESSED_OBJECTS column of the DBA_SQL_FIREWALL_SQL_LOGS view.
CURRENT_USER	VARCHAR2(128)		Name of the user who invoked the SQL command
TOP_LEVEL	VARCHAR2(9)		Indicates whether the allowed SQL entry is a top-level SQL command, that is, a SQL command issued directly from the user (Y) or a SQL command from a PL/SQL unit (N)
VERSION	NUMBER	NOT NULL	Version number for the allow-list when the allowed SQL was added You can use this value to determine whether specific allowed SQLs were added to the allow-list in the same batch.

Note:

This view is available starting with Oracle Database 23ai.

See Also:

"DBA_SQL_FIREWALL_SQL_LOGS"

A.6 DBA_SQL_FIREWALL_CAPTURE_LOGS

DBA_SQL_FIREWALL_CAPTURE_LOGS displays SQL Firewall capture logs.

Column	Datatype	NULL	Description
USERNAME	VARCHAR2(128)		Name of the target user
SESSION_ID	NUMBER	NOT NULL	SQL Firewall session ID
COMMAND_TYPE	VARCHAR2(64)		Type of SQL command
SQL_SIGNATURE	VARCHAR2(64)		SQL signature
SQL_TEXT	VARCHAR2(4000)		SQL text (up to 1000 characters) To view the full SQL text, query the SQL_TEXT column of the DBA_SQL_FIREWALL_SQL_LOGS view.

Column	Datatype	NULL	Description
ACCESSED_OBJECTS	VARCHAR2(4000)		List of accessed objects (up to 1000 characters) To view the full list of accessed objects, query the ACCESSED_OBJECTS column of the DBA_SQL_FIREWALL_SQL_LOGS view.
CURRENT_USER	VARCHAR2(128)		Name of the user who invoked the SQL command
TOP_LEVEL	VARCHAR2(9)		Indicates whether the SQL command is a top-level SQL command, that is, a SQL command issued directly from the user (Y) or a SQL command from a PL/SQL unit (N)
CLIENT_PROGRAM	VARCHAR2(84)		Name of the client program
OS_USER	VARCHAR2(128)		Name of the OS user of the client process
IP_ADDRESS	VARCHAR2(48)		Client IP address

Note:

This view is available starting with Oracle Database 23ai.

See Also:

"DBA_SQL_FIREWALL_SQL_LOGS"

A.7 DBA_SQL_FIREWALL_CAPTURES

DBA_SQL_FIREWALL_CAPTURES displays information about SQL Firewall captures.

Column	Datatype	NULL	Description
USERNAME	VARCHAR2(128)		Name of the target user
TOP_LEVEL_ONLY	VARCHAR2(14)		Indicates whether SQL Firewall should capture only top-level SQL commands, that is, SQL commands issued directly from the user (Y) or all SQL commands, including top-level SQL commands and SQL commands from PL/SQL units (N)
STATUS	VARCHAR2(8)		Capture status (ENABLED or DISABLED)
LAST_STARTED_ON	TIMESTAMP(6) WITH TIME ZONE		Date and time at which the capture was last started
LAST_STOPPED_ON	TIMESTAMP(6) WITH TIME ZONE		Date and time at which the capture was last stopped

Note:

This view is available starting with Oracle Database 23ai.

A.8 DBA_SQL_FIREWALL_SESSION_LOGS

DBA_SQL_FIREWALL_SESSION_LOGS displays SQL Firewall session logs.

Column	Datatype	NULL	Description
SESSION_ID	NUMBER	NOT NULL	SQL Firewall session ID
USERNAME	VARCHAR2(128)		Name of the target user
LOGIN_TIME	TIMESTAMP(6) WITH TIME ZONE		Date and time at which the target user logged in to the database
IP_ADDRESS	VARCHAR2(48)		Client IP address
CLIENT_PROGRAM	VARCHAR2(84)		Name of the client program
OS_USER	VARCHAR2(128)		Name of the OS user of the client process

> **Note:**
>
> This view is available starting with Oracle Database 23ai.

A.9 DBA_SQL_FIREWALL_SQL_LOGS

DBA_SQL_FIREWALL_SQL_LOGS displays information about SQL logs for SQL Firewall.

Column	Datatype	NULL	Description
COMMAND_TYPE	VARCHAR2(64)		Type of SQL command
SQL_SIGNATURE	VARCHAR2(64)		SQL signature
SQL_TEXT	CLOB		SQL text
ACCESSED_OBJECTS	CLOB		List of accessed objects
CHARSET	VARCHAR2(64)		Character set of the SQL text

> **Note:**
>
> This view is available starting with Oracle Database 23ai.

A.10 DBA_SQL_FIREWALL_STATUS

DBA_SQL_FIREWALL_STATUS displays the status of SQL Firewall.

Column	Datatype	NULL	Description
STATUS	VARCHAR2(8)		SQL Firewall status (ENABLED or DISABLED)

Column	Datatype	NULL	Description
STATUS_UPDATED_ON	TIMESTAMP(6) WITH TIME ZONE	NOT NULL	Date and time of the most recent SQL Firewall status change
EXCLUDE_JOBS	VARCHAR2(12)		Indicates whether the SQL Firewall will capture or enforce allow-lists for database connections and SQL executions of Oracle scheduler job sessions (Y) or not (N)

> **Note:**
>
> This view is available starting with Oracle Database 23ai.

A.11 DBA_SQL_FIREWALL_VIOLATIONS

DBA_SQL_FIREWALL_VIOLATIONS lists SQL Firewall violations.

Column	Datatype	NULL	Description
USERNAME	VARCHAR2(128)		Name of the target user
COMMAND_TYPE	VARCHAR2(64)		Type of SQL command
SQL_SIGNATURE	VARCHAR2(64)		SQL signature
SQL_TEXT	VARCHAR2(4000)		SQL text (up to 1000 characters) To view the full SQL text, query the SQL_TEXT column of the DBA_SQL_FIREWALL_SQL_LOGS view.
ACCESSED_OBJECTS	VARCHAR2(4000)		List of accessed objects (up to 1000 characters) To view the full list of accessed objects, query the ACCESSED_OBJECTS column of the DBA_SQL_FIREWALL_SQL_LOGS view.
CURRENT_USER	VARCHAR2(128)		Name of the user who invoked the SQL command
TOP_LEVEL	VARCHAR2(9)		Indicates whether the SQL command is a top-level SQL command, that is, a SQL command issued directly from the user (Y) or a SQL command from a PL/SQL unit (N)
IP_ADDRESS	VARCHAR2(48)		Client IP address
CLIENT_PROGRAM	VARCHAR2(84)		Name of the client program
OS_USER	VARCHAR2(128)		Name of the OS user of the client process
CAUSE	VARCHAR2(17)		Cause of the violation: • Context violation • SQL violation
FIREWALL_ACTION	VARCHAR2(7)		Indicates whether the SQL Firewall action was Allowed or Blocked
OCCURRED_AT	TIMESTAMP(6) WITH TIME ZONE		Date and time of the violation

Note:

This view is available starting with Oracle Database 23ai.

See Also:

"DBA_SQL_FIREWALL_SQL_LOGS"

B
DBMS_SQL_FIREWALL

The `DBMS_SQL_FIREWALL` package enables you to monitor users and detect or prevent SQL injection attacks against those users.

This chapter contains the following topics:

B.1 DBMS_SQL_FIREWALL Overview

The `DBMS_SQL_FIREWALL` PL/SQL package enables you to manage SQL Firewall, which tracks and can block SQL injection attacks.

The `DBMS_SQL_FIREWALL` package enables you to capture SQL activities of users, create allow-lists (that is, permitted actions) from the captured SQL activities, and then enforce the allow-lists to prevent or detect potential SQL injection attacks. In addition to SQL statements, the allow-list can contain a context list, which is a set of session contexts allowed for database connections. An example of a context can be IP addresses. You can also configure SQL Firewall to not run when Oracle Scheduler is running, because to do so may interfere with Oracle Scheduler operations. After you enable the allow-list, any SQL that the user performs will be monitored by SQL Firewall. SQL that the user performs that is not in the allow-list is considered to be a SQL injection attack. You can configure SQL Firewall to either allow the user to continue performing these SQL operations, or you can block these activities. Note that the SQL operations that violate the allow-list will always be written to a log table that you can query with data dictionary views.

You can configure SQL Firewall in both the root and in individual pluggable databases (PDBs).

Related Topics

- *Oracle Database Oracle SQL Firewall User's Guide*

B.2 DBMS_SQL_FIREWALL Security Model

Oracle Database protects the administration of SQL Firewall by storing its metadata in tables in the `SYS` schema.

Hence, these tables rely on dictionary protection, just as other dictionary tables in `SYS` do. Therefore, users who have the `SELECT ANY TABLE` system privilege cannot query these tables unless they also have the `SELECT ANY DICTIONARY` system privilege or are granted the `SELECT` object privileges on the tables. Only the `SYS` user can grant these privileges to other users.

Oracle Database stores the SQL Firewall tables in the `SYSAUX` tablespace by default. If you want to move the SQL Firewall log tables to a different (user-defined) tablespace, then you must first disable SQL Firewall, and then use the `MOVE` clause of the `ALTER TABLE` statement to perform the move operation.

To use the procedures in the DBMS_SQL_FIREWALL package, a user must be granted the SQL_FIREWALL_ADMIN role.

Related Topics

- *Oracle Database Oracle SQL Firewall User's Guide*

B.3 DBMS_SQL_FIREWALL Constants

The DBMS_SQL_FIREWALL package provides constants that are used with several SQL Firewall procedures.

These constants are described in the following table.

Table B-1 DBMS_SQL_FIREWALL Constants

Name	Type	Value	Description
DBMS_SQL_FIREWALL.ENFORCE_ALL	NUMBER	3	Enforces both allowed SQL and allowed contexts when you run the DBMS_SQL_FIREWALL.ENABLE_ALLOW_LIST procedure
DBMS_SQL_FIREWALL.ENFORCE_CONTEXT	NUMBER	1	Enforces allowed contexts when you run the DBMS_SQL_FIREWALL.ENABLE_ALLOW_LIST procedure.
DBMS_SQL_FIREWALL.ENFORCE_SQL	NUMBER	2	Enforces allowed SQL when you run the DBMS_SQL_FIREWALL.ENABLE_ALLOW_LIST procedure
DBMS_SQL_FIREWALL.ALL_LOGS	NUMBER	3	Purges all logs when you run the DBMS_SQL_FIREWALL.PURGE procedure
DBMS_SQL_FIREWALL.CAPTURE_LOG	NUMBER	1	Purges only capture logs when you run the DBMS_SQL_FIREWALL.PURGE procedure
DBMS_SQL_FIREWALL.IP_ADDRESS	NUMBER	3	Specifies the user's IP address when you run the DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT or DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT procedure
DBMS_SQL_FIREWALL.OS_PROGRAM	NUMBER	1	Specifies the user's operating system program when you run the DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT or DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT procedure

Table B-1 (Cont.) DBMS_SQL_FIREWALL Constants

Name	Type	Value	Description
DBMS_SQL_FIREWALL.OS_USERNAME	NUMBER	2	Specifies an operating system name when you run the DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT or DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT procedure
DBMS_SQL_FIREWALL.SCHEDULER_JOB	NUMBER	1	Indicates whether SQL Firewall will capture and enforce allow-lists for database connections and SQL executions during Oracle Scheduler operations. Use this constant with the DBMS_SQL_FIREWALL.EXCLUDE and DBMS_SQL_FIREWALL.INCLUDE procedures.
DBMS_SQL_FIREWALL.VIOLATION_LOG	NUMBER	2	Purges only violation logs when you run the DBMS_SQL_FIREWALL.PURGE procedure

B.4 Summary of DBMS_SQL_FIREWALL Subprograms

This table lists and describes the DBMS_SQL_FIREWALL package subprograms.

Table B-2 DBMS_SQL_FIREWALL Package Subprograms

Subprogram	Description
ADD_ALLOWED_CONTEXT Procedure	Adds a context to the list of allowed contexts for a user who is configured for SQL Firewall
APPEND_ALLOW_LIST Procedure	Appends additional contents to an existing allow-list by using the existing capture logs or violation logs of the user, or both
APPEND_ALLOW_LIST_SINGLE_SQL Procedure	Appends a single SQL record to the violation log or capture log to an existing allow-list
CREATE_CAPTURE Procedure	Creates a SQL Firewall capture for a specified user at a given level
DELETE_ALLOWED_CONTEXT Procedure	Deletes a SQL Firewall context value that had been assigned to a user
DELETE_ALLOWED_SQL Procedure	Deletes a specified entry from the allowed SQL that had been assigned to a user
DISABLE Procedure	Disables SQL Firewall
DISABLE_ALLOW_LIST Procedure	Disables SQL Firewall allow-list enforcement for a given user
DROP_ALLOW_LIST Procedure	Deletes the SQL Firewall allow-list of a specified user

Table B-2 (Cont.) DBMS_SQL_FIREWALL Package Subprograms

Subprogram	Description
DROP_CAPTURE Procedure	Drops a SQL Firewall capture and deletes all the associated capture logs
ENABLE Procedure	Enables SQL Firewall
ENABLE_ALLOW_LIST Procedure	Enables SQL Firewall allow-list enforcement for a given user
EXCLUDE Procedure	Prevents SQL Firewall from capturing or enforcing allow-lists for database connections and SQL executions during Oracle Scheduler operations
EXPORT_ALLOW_LIST Procedure	Exports the allow-list of the given user in JSON format, into the CLOB provided from the `allow_list` argument
FLUSH_LOGS Procedure	Flushes all the SQL Firewall logs that reside in the memory into the log tables
GENERATE_ALLOW_LIST Procedure	Generates a SQL Firewall allow-list for the specified user by using data from the existing capture logs of the user
IMPORT_ALLOW_LIST Procedure	Imports the allow-list from the specified `CLOB` for the given user, to the target database
INCLUDE Procedure	Enables SQL Firewall to capture and enforce allow-lists for database connections and SQL executions during Oracle Scheduler operations
PURGE_LOG Procedure	Purges SQL Firewall logs
START_CAPTURE Procedure	Starts a SQL Firewall capture for a user
STOP_CAPTURE Procedure	Stops a SQL Firewall capture for a user
UPDATE_ALLOW_LIST_ENFORCEMENT Procedure	Updates the SQL Firewall allow-list enforcement options for the given user

B.4.1 ADD_ALLOWED_CONTEXT Procedure

This procedure adds a context to the list of allowed contexts for a user's SQL Firewall allow-list.

Syntax

```
DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT (
   username       IN  VARCHAR2,
   context_type   IN  NUMBER,
   value          IN  VARCHAR2);
```

Parameters

Table B-3 ADD_ALLOWED_CONTEXT Procedure Parameters

Parameter	Description
`username`	Specifies the name of the user who has a SQL Firewall allow-list. To find all the users who has an allow-list, query `DBA_SQL_FIREWALL_ALLOW_LISTS`.

Table B-3 (Cont.) ADD_ALLOWED_CONTEXT Procedure Parameters

Parameter	Description
context_type	Specifies one of the following context types: • DBMS_SQL_FIREWALL.IP_ADDRESS accepts IPv4 and IPv6 addresses and subnets in the CIDR notation. • DBMS_SQL_FIREWALL.OS_USERNAME accepts any valid operating system user name, such as oracle. • DBMS_SQL_FIREWALL.OS_PROGRAM accepts any valid operating system program name, such as sqlplus or SQL Developer.
value	Specifies the value of the context_type constant, such as an IP address for DBMS_SQL_FIREWALL.IP_ADDRESS. To allow a local (bequeathed) connection that does not have an IP address, specify with the value Local for the DBMS_SQL_FIREWALL.IP_ADDRESS type. To specify all values of the context (such as all possible operating system programs), then enter the % wild card character.

Usage Notes

- You can find the user's current context type settings by querying the following data dictionary views:
  - DBA_SQL_FIREWALL_ALLOWED_IP_ADDR
  - DBA_SQL_FIREWALL_ALLOWED_OS_PROG
  - DBA_SQL_FIREWALL_ALLOWED_OS_USER
- Before you can add any contexts for the user, the user's allow-list must be created (using the DBMS_SQL_FIREWALL.GENERATE_ALLOW_LIST procedure).
- This procedure can be run when the allow-list is enabled or disabled, and it takes effects immediately.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.ADD_ALLOWED_CONTEXT (
    username       => 'PFITCH',
    context_type   => DBMS_SQL_FIREWALL.OS_PROGRAM,
    value          => 'SQL Developer'
   );
END;
/
```

B.4.2 APPEND_ALLOW_LIST Procedure

This procedure appends additional contents to an existing allow-list by using the existing capture logs or violation logs of the user, or both.

Syntax

```
DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST (
   username       IN  VARCHAR2,
   source         IN  NUMBER);
```

Parameters

Table B-4 APPEND_ALLOW_LIST Procedure Parameters

Parameter	Description
username	Specifies the name of the user who was designated for the SQL Firewall allow-list. To find this user, query DBA_SQL_FIREWALL_ALLOW_LISTS.
source	Specifies one of the following log types: • DBMS_SQL_FIREWALL.CAPTURE_LOG • DBMS_SQL_FIREWALL.VIOLATION_LOG • DBMS_SQL_FIREWALL.ALL_LOGS

Usage Notes

- DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST processes the specified source logs and identifies contents to be appended to the allow-list. Then it populates the SQL Firewall metadata tables for the allowed SQL and allowed contexts, which will be used during the allow-list enforcement.
- You can run this procedure when the allow-list is either enabled or disabled.
- The change takes effect immediately.
- A new allow-list version number will be associated with all the allowed SQL entries added by the same DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST execution. This new version number will be 1 plus the current maximum allow-list version of the specified user.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST (
    username       => 'PFITCH',
    source         => DBMS_SQL_FIREWALL.CAPTURE_LOG
   );
END;
/
```

B.4.3 APPEND_ALLOW_LIST_SINGLE_SQL Procedure

This procedure appends a single SQL record to the violation log or capture log to an existing allow-list.

This procedure is useful for when you want to individually append SQL commands from the violations log or the capture log to an existing allow-list.

Syntax

```
DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL (
   username       IN  VARCHAR2,
   sql_signature  IN  VARCHAR2,
   current_user   IN  VARCHAR2,
   top_level      IN  VARCHAR2,
   source         IN  NUMBER DEFAULT);
```

Parameters

Table B-5 APPEND_ALLOW_LIST_SINGLE_SQL Procedure Parameters

Parameter	Description
username	Specifies the name of the user who was designated for the SQL Firewall allow-list. To find this user, query DBA_SQL_FIREWALL_ALLOW_LISTS.
sql_signature	Specifies the signature of the SQL to be added. To find the signature of the SQL for the target record, query the DBA_SQL_FIREWALL_CAPTURE or DBA_SQL_FIREWALL_VIOLATIONS dynamic view.
current_user	Specifies the name of the user who the SQL command was executed as. For example, if user pfitch invokes a definer's rights procedure created in the psmith schema, then all the SQL commands in the procedure are executed as psmith, the current_user. If the procedure is an invoker's rights procedure, then the current_user is the invoker, pfitch.
top_level	Specifies whether the SQL that was executed was top level. Possible values are as follows: • Y (for Yes) means that the target SQL record is top-level (that is, the statement that the user directly runs). • N (for No) means that the target SQL record is not top-level (that is, the SQL command that is issued from PL/SQL units).
source	Specifies the source log to add the SQL record from: • DBMS_SQL_FIREWALL.CAPTURE_LOG • DBMS_SQL_FIREWALL.VIOLATION_LOG (default)

Usage Notes

- DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL processes the specified source log and identifies the target SQL record to be appended to the allow-list. Then it populates the SQL Firewall metadata tables for the allowed SQL, which will be used during the allow-list enforcement.
- You can run this procedure when the allow-list is either enabled or disabled.
- The change takes effect immediately.
- A new allow-list version number will be associated with the newly added allowed SQL entry.

Example

1. Query the DBA_SQL_FIREWALL_VIOLATIONS or the DBA_SQL_FIREWALL_CAPTURE_LOGS data dictionary view to find the target SQL record that you want to add to the allow-list. Obtain the values for the USERNAME, SQL_SIGNATURE, CURRENT_USER, and TOP_LEVEL columns for the target SQL record.
2. Enter these values in the DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL SQL procedure to add the target SQL record to the allow-list.
   For example:

```
BEGIN
  DBMS_SQL_FIREWALL.APPEND_ALLOW_LIST_SINGLE_SQL (
    username       => 'PFITCH',
    sql_signature  =>
'7D33A84D0A1B56E382B9A92D01BCD19933969CB16E2AB4934A2258563F5ADB44',
```

```
        current_user   => 'PSMITH',
        top_level      => 'N',
        source         => DBMS_SQL_FIREWALL.CAPTURE_LOG
       );
    END;
    /
```

B.4.4 CREATE_CAPTURE Procedure

This procedure creates a SQL Firewall capture for a specified user at a given level.

Syntax

```
DBMS_SQL_FIREWALL.CREATE_CAPTURE (
   username       IN VARCHAR2,
   top_level_only IN BOOLEAN,
   start_capture  IN BOOLEAN);
```

Parameters

Table B-6 CREATE_CAPTURE Procedure Parameters

Parameter	Description
username	Specifies the name of the user whose SQL Firewall capture is to be created. To find existing users, query DBA_SQL_FIREWALL_CAPTURES.
top_level_only	• TRUE captures only SQL statements that have been directly issued by the user • FALSE captures both top-level SQL statements and SQL statements that have been issued by PL/SQL units. This setting is the default.
start_capture	• TRUE starts the capture process right away, after you run DBMS_SQL_FIREWALL.CREATE_CAPTURE. This setting is the default. • FALSE does not start the capture process. You can start it later on by using DBMS_SQL_FIREWALL.START_CAPTURE.

Usage Notes

To find the status of existing SQL Firewall captures, including users who have already been configured for SQL Firewall captures, query the DBA_SQL_FIREWALL_CAPTURES data dictionary view.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.CREATE_CAPTURE (
    username         => 'C##HR_ADMIN',
    top_level_only   => TRUE,
    start_capture    => TRUE
  );
END;
/
```

B.4.5 DELETE_ALLOWED_CONTEXT Procedure

This procedure deletes a context from the list of allowed contexts for a user's SQL Firewall allow-list.

Syntax

```
DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT (
   username       IN  VARCHAR2,
   context_type   IN  NUMBER,
   value          IN  VARCHAR2);
```

Parameters

Table B-7 DELETE_ALLOWED_CONTEXT Procedure Parameters

Parameter	Description
username	Specifies the name of the user who was designated for the SQL Firewall allow-list. To find this user, query DBA_SQL_FIREWALL_ALLOW_LISTS.
context_type	• DBMS_SQL_FIREWALL.IP_ADDRESS accepts IPv4 and IPv6 addresses and subnets in the CIDR notation. • DBMS_SQL_FIREWALL.OS_USERNAME accepts any valid operating system user name, such as oracle. • DBMS_SQL_FIREWALL.OS_PROGRAM accepts any valid operating system program name, such as sqlplus or SQL Developer.
value	Specifies the value of the context_type constant, such as an IP address for DBMS_SQL_FIREWALL.IP_ADDRESS. If you omit this value or specify NULL, then all the allowed context values of the specified context type are deleted. This setting is the default.

Usage Notes

- You can find the user's current context type settings by querying the following data dictionary views:
  - DBA_SQL_FIREWALL_ALLOWED_IP_ADDR
  - DBA_SQL_FIREWALL_ALLOWED_OS_PROG
  - DBA_SQL_FIREWALL_ALLOWED_OS_USER
- This procedure can be run when the allow-list is enabled or disabled, and it takes effects immediately.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.DELETE_ALLOWED_CONTEXT, (
    username       => 'PFITCH',
    context_type   => DBMS_SQL_FIREWALL.OS_PROGRAM,
    value          => 'SQL Developer'
   );
END;
/
```

B.4.6 DELETE_ALLOWED_SQL Procedure

This procedure deletes a specified entry from the list of allowed SQL for a user's SQL Firewall allow-list

Syntax

```
DBMS_SQL_FIREWALL.DELETE_ALLOWED_SQL (
   username       IN  VARCHAR2,
   allowed_sql_id IN  NUMBER);
```

Parameters

Table B-8 DELETE_ALLOWED_SQL Procedure Parameters

Parameter	Description
`username`	Specifies the name of the user who was designated for the SQL Firewall allow-list. To find this user, query `DBA_SQL_FIREWALL_ALLOW_LISTS`.
`allowed_sql_id`	Specifies the ID of the allowed SQL entry to be deleted from the allowed SQL of this user.To find this value, query `DBA_SQL_FIREWALL_ALLOWED_SQL`.

Usage Notes

- You can run this procedure when the allow-list is either enabled or disabled.
- The change takes effect immediately.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.DELETE_ALLOWED_SQL (
    username         => 'PFITCH',
    allowed_sql_id   => 1
   );
END;
/
```

B.4.7 DISABLE Procedure

This procedure disables SQL Firewall and stops all the existing captures and allow-lists that are enabled.

Syntax

```
DBMS_SQL_FIREWALL.DISABLE;
```

Parameters

None

Usage Notes

You can find the current status of SQL Firewall by querying the `DBA_SQL_FIREWALL_STATUS` data dictionary view.

Example

```
EXEC DBMS_SQL_FIREWALL.DISABLE;
```

B.4.8 DISABLE_ALLOW_LIST Procedure

This procedure immediately disables SQL Firewall allow-list enforcement for a given user.

Syntax

```
DBMS_SQL_FIREWALL.DISABLE_ALLOW_LIST (
   username       IN  VARCHAR2);
```

Parameters

Table B-9 DISABLE_ALLOW_LIST Procedure Parameters

Parameter	Description
username	Specifies the name of the user who was designated for the SQL Firewall allow-list. To find this user, query DBA_SQL_FIREWALL_ALLOW_LISTS. If you specify NULL, then all allow-lists that are currently enabled will be disabled.

Usage Notes

To find the status of users' allow-lists, query the DBA_SQL_FIREWALL_ALLOW_LISTS data dictionary view.

Example

```
EXEC DBMS_SQL_FIREWALL.DISABLE_ALLOW_LIST ('PFITCH');
```

B.4.9 DROP_ALLOW_LIST Procedure

This procedure deletes the SQL Firewall allow-list of a specified user.

Syntax

```
DBMS_SQL_FIREWALL.DROP_ALLOW_LIST (
   username       IN  VARCHAR2);
```

Parameters

Table B-10 DROP_ALLOW_LIST Procedure Parameters

Parameter	Description
username	Specifies the name of the user who was designated for the SQL Firewall allow-list. To find this user, query DBA_SQL_FIREWALL_ALLOW_LISTS.

Usage Notes

- To find the status of users' allow-lists, query the DBA_SQL_FIREWALL_ALLOW_LISTS data dictionary view.

- You cannot drop an allow-list that is currently enabled. To disable an allow-list, run the `DBMS_SQL_FIREWALL.DISABLE_ALLOW_LIST` procedure.

Example

```
EXEC DBMS_SQL_FIREWALL.DROP_ALLOW_LIST ('PFITCH');
```

B.4.10 DROP_CAPTURE Procedure

This procedure drops a SQL Firewall capture and deletes all the associated capture logs.

Syntax

```
DBMS_SQL_FIREWALL.DROP_CAPTURE (
   username       IN VARCHAR2);
```

Parameters

Table B-11 DROP_CAPTURE procedure Parameters

Parameter	Description
`username`	Specifies the name of the user whose SQL Firewall capture is to be dropped. To find this user, query `DBA_SQL_FIREWALL_CAPTURES`.

Usage Notes

- To find the status of existing SQL Firewall captures, query the `DBA_SQL_FIREWALL_CAPTURES` data dictionary view.
- You cannot drop a capture that is currently running. To stop the capture, run the `DBMS_SQL_FIREWALL.STOP_CAPTURE` procedure.
- Dropping a capture for a user does not affect the user's allow-list, which can continue to run even if the capture has been dropped. Captures and allow-lists are separate entities.

Example

```
EXEC DBMS_SQL_FIREWALL.DROP_CAPTURE ('C##HR_ADMIN');
```

B.4.11 ENABLE Procedure

This procedure enables SQL Firewall and starts all existing captures and allow-lists that are configured to be enabled.

Syntax

```
DBMS_SQL_FIREWALL.ENABLE;
```

Parameters

None

Usage Notes

You can find the current status of SQL Firewall by querying the `DBA_SQL_FIREWALL_STATUS` data dictionary view.

Example

```
EXEC DBMS_SQL_FIREWALL.ENABLE;
```

B.4.12 ENABLE_ALLOW_LIST Procedure

This procedure immediately enables SQL Firewall allow-list enforcement for a given user.

Syntax

```
DBMS_SQL_FIREWALL.ENABLE_ALLOW_LIST (
   username       IN  VARCHAR2,
   enforce        IN  NUMBER,
   block          IN  BOOLEAN;
```

Parameters

Table B-12 ENABLE_ALLOW_LIST Procedure Parameters

Parameter	Description
`username`	Specifies the name of the user whose SQL Firewall allow-list is to be enabled. To find this user, query `DBA_SQL_FIREWALL_ALLOW_LISTS`. If you enter `NULL`, then the allow-lists for all users who do not yet have allow-lists enabled are enabled.
`enforce`	• `DBMS_SQL_FIREWALL.ENFORCE_CONTEXT` enforces the allowed contexts that have been configured. • `DBMS_SQL_FIREWALL.ENFORCE_SQL` enforces the allowed SQL that has been configured. • `DBMS_SQL_FIREWALL.ENFORCE_ALL` enforces both allowed contexts and allowed SQL. This setting is the default.
`block`	• `TRUE` blocks user's database connection or the user's SQL execution whenever the user violates the allow-list definition. • `FALSE` allows unmatched user database connections or SQL commands to proceed. This setting is the default.

Usage Notes

- To find the status of users' allow-lists, query the `DBA_SQL_FIREWALL_ALLOW_LISTS` data dictionary view.
- SQL Firewall always generates a violation log for any unmatched database connection or SQL statement regardless of the `block` option setting.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.ENABLE_ALLOW_LIST (
    username       => 'PFITCH',
    enforce        => DBMS_SQL_FIREWALL.ENFORCE_SQL,
```

```
    block          => TRUE
   );
END;
/
```

B.4.13 EXCLUDE Procedure

This procedure prevents SQL Firewall from capturing or enforcing allow-lists for database connections and SQL executions during Oracle Scheduler operations.

Oracle Scheduler jobs are often used in databases for various maintenance purposes. Accidentally interrupting critical jobs can cause undesirable consequences. You can configure SQL Firewall to not capture any SQL statements nor enforce any allow-lists that are run during an Oracle Scheduler job session. This procedure applies to all users that have been configured for SQL Firewall captures and allow-lists. By default, Oracle Scheduler jobs are excluded from SQL Firewall operations.

Syntax

```
DBMS_SQL_FIREWALL.EXCLUDE (
   FEATURE        IN NUMBER);
```

Parameters

Table B-13 EXCLUDE Procedure Parameters

Parameter	Description
FEATURE	Enter DBMS_SQL_FIREWALL.SCHEDULER_JOB for this value.

Usage Notes

- To find the status of whether SQL Firewall is enforced during Oracle Scheduler operations, query the EXCLUDE_JOBS column of the DBA_SQL_FIREWALL_STATUS data dictionary view. If the output is Y, then Oracle Scheduler jobs are excluded from SQL Firewall operations.
- To enable Oracle Firewall to run during Oracle Scheduler operations, run the DBMS_SQL_FIREWALL.INCLUDE procedure.

Example

```
EXEC DBMS_SQL_FIREWALL.EXCLUDE (DBMS_SQL_FIREWALL.SCHEDULER_JOB);
```

B.4.14 EXPORT_ALLOW_LIST Procedure

This procedure exports the allow-list of the given user in JSON format, into the CLOB provided from the allow_list argument.

Syntax

```
DBMS_SQL_FIREWALL.EXPORT_ALLOW_LIST (
   username       IN      VARCHAR2,
   allow_list     IN/OUT  CLOB;
```

Parameters

Table B-14 EXPORT_ALLOW_LIST Procedure Parameters

Parameter	Description
`username`	Specifies the user that the allow-list was created for. To find which user has an allow-list, query `DBA_SQL_FIREWALL_ALLOW_LISTS`.
`allow_list`	Specifies the `CLOB` (which must already exist) into which the exported allow-list must go

Usage Notes

- Before you run this procedure, you must create the `CLOB` and then pass it to the API (for example, by `DBMS_LOB.CREATETEMPORARY` for the PL/SQL client, or by `OracleConnection.createClob()` for JDBC Java client).
- The export operation includes the allow-list's settings (`status`, `enforce`, `block`, `top_level_only`, `generated_on`, and `status_updated_on timestamp`), allowed SQL, and allowed contexts. In addition, the export operation includes all the referenced SQL logs (by the allowed SQL).
- `DBMS_SQL_FIREWALL.EXPORT_ALLOW_LIST` does not export capture logs or violation logs.
- To find the status of users' allow-lists, query the `DBA_SQL_FIREWALL_ALLOW_LISTS` data dictionary view.
- If you want to export all the SQL Firewall metadata, which includes captures and allow-lists for all users, then instead of using `DBMS_SQL_FIREWALL.EXPORT_ALLOW_LIST`, use the `include=SQL_FIREWALL` clause in the Oracle Data Pump `expdp` command. See *Oracle Database Security Guide*.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.EXPORT_ALLOW_LIST (
    username       => 'PFITCH',
    allow_list     => ALLOW_LIST_CLOB;
   );
END;
/
```

B.4.15 FLUSH_LOGS Procedure

This procedure flushes all the SQL Firewall logs that reside in the memory into the log tables.

Syntax

```
DBMS_SQL_FIREWALL.FLUSH_LOGS;
```

Parameters

None

Usage Notes

- Usually you do not need to invoke this procedure explicitly, because logs in the memory are flushed to the log tables frequently in the background. But in case if you want to see the capture logs or violation logs immediately after the action during when SQL Firewall is running, you can run this procedure before looking at the logs.
- The `DBMS_SQL_FIREWALL.FLUSH_LOGS` procedure is equivalent to the `DBMS_MEMOPTIMIZE_ADMIN.WRITES_FLUSH` procedure. (See WRITES_FLUSH Procedure.)

Example

```
EXEC DBMS_SQL_FIREWALL.FLUSH_LOGS;
```

B.4.16 GENERATE_ALLOW_LIST Procedure

This procedure generates a SQL Firewall allow-list for the specified user by using the existing capture logs of the user.

Syntax

```
DBMS_SQL_FIREWALL.GENERATE_ALLOW_LIST (
   username       IN  VARCHAR2;
```

Parameters

Table B-15 GENERATE_ALLOW_LIST Procedure Parameters

Parameter	Description
`username`	Specifies the name of the user who was designated for the SQL Firewall allow-list. To find this user, query `DBA_SQL_FIREWALL_CAPTURES`.

Usage Notes

- To find information about existing generated allow-lists, query the `DBA_SQL_FIREWALL_ALLOW_LISTS` data dictionary view.
- Before you run this procedure, the following components must be in place:
  - The specified user must exist.
  - A capture (using `DBMS_SQL_FIREWALL.CREATE_CAPTURE`) has been created for this user. This capture must be disabled (using `DBMS_SQL_FIREWALL.STOP_CAPTURE`) before you can generate an allow-list for the user.
  - No allow-list exists yet for the user.

Example

```
EXEC DBMS_SQL_FIREWALL.GENERATE_ALLOW_LIST ('PFITCH');
```

B.4.17 IMPORT_ALLOW_LIST Procedure

This procedure imports the allow-list from the specified `CLOB` for the given user, to the target database.

Syntax

```
DBMS_SQL_FIREWALL.IMPORT_ALLOW_LIST (
   username       IN      VARCHAR2,
   allow_list     IN      CLOB;
```

Parameters

Table B-16 IMPORT_ALLOW_LIST Procedure Parameters

Parameter	Description
`username`	Specifies the user of the exported allow-list. To check whether this user already had an allow-list created in the target database, query `DBA_SQL_FIREWALL_ALLOW_LISTS`.
`allow_list`	Specifies the `CLOB` that was created when the allow-list was exported with `DBMS_SQL_FIREWALL.EXPORT_ALLOW_LIST`.

Usage Notes

- If this user does not have an allow-list in the target database, a new allow-list will be created for this user using the allow-list from the JSON payload. The new allow-list will have the same settings (`status`, `top_level_only`, `enforce`, `block`, `generated_on`, `status_updated_on`), same allowed contexts and same allowed SQL as the one in the JSON. If the specified user already has an allow-list in the target database, then all the settings (`status`, `top_level_only`, `enforce`, `block`, and various timestamps) of the existing allow-list will remain untouched, but only the allowed SQL and allowed contexts from the JSON will be merged into the ones for the existing allow-list.
- In addition, the import operation includes all the referenced SQL logs (by the allowed SQL).
- To find the status of users' allow-lists, query the `DBA_SQL_FIREWALL_ALLOW_LISTS` data dictionary view.
- If you want to import all the SQL Firewall metadata, which includes captures and allow-lists, then instead of using `DBMS_SQL_FIREWALL.IMPORT_ALLOW_LIST`, use the `include=SQL_FIREWALL` clause in the Oracle Data Pump `impdp` command. See *Oracle Database Security Guide*.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.IMPORT_ALLOW_LIST (
    username       => 'PFITCH',
    allow_list     => ALLOW_LIST_CLOB;
   );
END;
/
```

B.4.18 INCLUDE Procedure

This procedure enables SQL Firewall to capture and enforce allow-lists for database connections and SQL executions during Oracle Scheduler operations.

Syntax

```
DBMS_SQL_FIREWALL.INCLUDE (
   FEATURE        IN NUMBER);
```

Parameters

Table B-17 INCLUDE Procedure Parameters

Parameter	Description
FEATURE	Enter DBMS_SQL_FIREWALL.SCHEDULER_JOB for this value.

Usage Notes

- To find the status of whether SQL Firewall is enforced during Oracle Scheduler operations, query the EXCLUDE_JOBS column of the DBA_SQL_FIREWALL_STATUS data dictionary view. If the output is N, then SQL Firewall can perform during Oracle Scheduler operations.
- To prevent SQL Firewall from running during Oracle Scheduler operations, run the DBMS_SQL_FIREWALL.EXCLUDE procedure.

Example

```
EXEC DBMS_SQL_FIREWALL.INCLUDE (DBMS_SQL_FIREWALL.SCHEDULER_JOB);
```

B.4.19 PURGE_LOG Procedure

This procedure purges SQL Firewall logs that belong to the given user based on the specified purge time (that is, logs that were generated before the specified purge time).

Syntax

```
BEGIN
  DBMS_SQL_FIREWALL.PURGE_LOG (
   username       IN  VARCHAR2,
   purge_time     IN  TIMESTAMP WITH TIME ZONE,
   log_type       IN  NUMBER);
```

Parameters

Table B-18 PURGE_LOG Procedure Parameters

Parameter	Description
username	Specifies the user whose capture logs or violation logs you want to purge. To see capture logs, query DBA_SQL_FIREWALL_CAPTURE_LOGS; to see violation logs, query DBA_SQL_FIREWALL_VIOLATIONS.

Table B-18 (Cont.) PURGE_LOG Procedure Parameters

Parameter	Description
purge_time	The timestamp (in TIMESTAMP format) that you can specify to purge only logs that were generated before a certain time. If you omit this value, then Oracle Database purges all logs, regardless of the time when they were generated.
log_type	Specifies the type of the logs to be purged. • DBMS_SQL_FIREWALL.CAPTURE_LOG • DBMS_SQL_FIREWALL.VIOLATION_LOG • DBMS_SQL_FIREWALL.ALL_LOGS (default)

Usage Notes

To find information about SQL Firewall logs, query the DBA_SQL_FIREWALL_VIOLATIONS data dictionary view.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.PURGE_LOG (
    username     => 'PSMITH',
    purge_time   => TO_TIMESTAMP_TZ('23-JAN-22 18.44.42 -07:00', 'DD/MM/YY
HH24:MI:SS TZH:TZM'),
    log_type     => DBMS_SQL_FIREWALL.VIOLATION_LOG
   );
END;
/
```

B.4.20 START_CAPTURE Procedure

This procedure immediately starts a SQL Firewall capture for a user.

Syntax

```
DBMS_SQL_FIREWALL.START_CAPTURE (
   username       IN  VARCHAR2);
```

Parameters

Table B-19 START_CAPTURE Procedure Parameters

Parameter	Description
username	Specifies the name of the user to be designated for the SQL Firewall capture.

Usage Notes

- A user can only have one SQL Firewall capture. To find if the user already has been configured for a capture, query the DBA_SQL_FIREWALL_CAPTURES data dictionary view.
- After you start the capture process, all SQL the user enters is captured into the SQL Firewall capture log table. You can periodically check the this SQL by querying the DBA_SQL_FIREWALL_CAPTURE_LOGS data dictionary view.

Example

```
EXEC DBMS_SQL_FIREWALL.START_CAPTURE ('PFITCH');
```

B.4.21 STOP_CAPTURE Procedure

This procedure immediately stops a SQL Firewall capture for a given user.

Syntax

```
DBMS_SQL_FIREWALL.STOP_CAPTURE (
   username       IN  VARCHAR2);
```

Parameters

Table B-20 STOP_CAPTURE Procedure Parameters

Parameter	Description
username	Specifies the name of the user who was designated for the SQL Firewall capture. To find this user, query DBA_SQL_FIREWALL_CAPTURES.

Usage Notes

- The capture process must be currently running before you can run this procedure. You can check its status by querying the DBA_SQL_FIREWALL_CAPTURES data dictionary view.
- After you stop the capture process, you can generate an allow-list for the user by running the DBMS_SQL_FIREWALL.GENERATE_ALLOW_LIST procedure.

Example

```
EXEC DBMS_SQL_FIREWALL.STOP_CAPTURE ('PFITCH');
```

B.4.22 UPDATE_ALLOW_LIST_ENFORCEMENT Procedure

This procedure immediately updates the SQL Firewall allow-list enforcement options for the given user.

Syntax

```
BEGIN
  DBMS_SQL_FIREWALL.UPDATE_ALLOW_LIST_ENFORCEMENT (
   username       IN  VARCHAR2,
   enforce        IN  NUMBER,
   block          IN  BOOLEAN);
```

Parameters

Table B-21 UPDATE_ALLOW_LIST_ENFORCEMENT Procedure Parameters

Parameter	Description
username	Specifies the name of the user for whom the allow-list was generated. To find this user, query DBA_SQL_FIREWALL_ALLOW_LISTS. If you enter NULL, then the enforcement options of all the existing allow-lists (both enabled or disabled allow-lists) are updated.
enforce	• DBMS_SQL_FIREWALL.ENFORCE_CONTEXT enforces the allowed contexts that have been configured. • DBMS_SQL_FIREWALL.ENFORCE_SQL enforces the allowed SQL that has been configured. • DBMS_SQL_FIREWALL.ENFORCE_ALL enforces both allowed contexts and allowed SQL. This setting is the default.
block	• TRUE blocks user's database connection or the user's SQL execution whenever the user violates the allow-list definition. • FALSE allows unmatched user database connections or SQL commands to proceed. This setting is the default.

Usage Notes

To find the status of users' allow-lists, query the DBA_SQL_FIREWALL_ALLOW_LISTS data dictionary view.

Example

```
BEGIN
  DBMS_SQL_FIREWALL.UPDATE_ALLOW_LIST_ENFORCEMENT (
    username       => 'PFITCH',
    enforce        => DBMS_SQL_FIREWALL.ENFORCE_SQL,
    block          => TRUE
   );
END;
/
```