



Extreme Networks Security Ariel Query Language Guide

Copyright © 2013–2015 All rights reserved.

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, please see:

www.extremenetworks.com/company/legal/trademarks/

Support

For product support, including documentation, visit: www.extremenetworks.com/documentation/

For information, contact:

Extreme Networks, Inc.
145 Rio Robles
San Jose, California 95134
USA

Table of Contents

About this guide.....	4
Conventions.....	4
Providing Feedback to Us.....	5
Getting Help.....	6
Related Publications.....	6
Chapter 1: Ariel Query Language (AQL).....	8
Ariel Query Language (AQL) deprecated versions.....	8
AQL functions.....	11
Logical and comparative operators.....	15
Event, flow and simarc fields for AQL queries.....	18
SELECT statement.....	22
WHERE clause.....	23
GROUP BY clause.....	24
ORDER BY clause.....	26
LIKE clause.....	27
COUNT function.....	28
Index.....	29



About this guide

The Ariel Query Language (AQL) Guide provides you with information for using the AQL advanced searching and API.

Intended audience

System administrators who view event or flow data stored in the Ariel database.

Statement of good security practices

IT system security involves protecting systems and information through prevention, detection and response to improper access from within and outside your enterprise. Improper access can result in information being altered, destroyed, misappropriated or misused or can result in damage to or misuse of your systems, including for use in attacks on others. No IT system or product should be considered completely secure and no single product, service or security measure can be completely effective in preventing improper use or access. Extreme Networks® systems, products and services are designed to be part of a lawful comprehensive security approach, which will necessarily involve additional operational procedures, and may require other systems, products or services to be most effective. EXTREME NETWORKS DOES NOT WARRANT THAT ANY SYSTEMS, PRODUCTS OR SERVICES ARE IMMUNE FROM, OR WILL MAKE YOUR ENTERPRISE IMMUNE FROM, THE MALICIOUS OR ILLEGAL CONDUCT OF ANY PARTY.

Note



Use of this Program may implicate various laws or regulations, including those related to privacy, data protection, employment, and electronic communications and storage. Extreme Networks Security Analytics may be used only for lawful purposes and in a lawful manner. Customer agrees to use this Program pursuant to, and assumes all responsibility for complying with, applicable laws, regulations and policies. Licensee represents that it will obtain or has obtained any consents, permissions, or licenses required to enable its lawful use of Extreme Networks Security Analytics.

Conventions

This section discusses the conventions used in this guide.

Text Conventions

The following tables list text conventions that are used throughout this guide.

Table 1: Notice Icons





Icon	Notice Type	Alerts you to...
	Note	Important features or instructions.
	Caution	Risk of personal injury, system damage, or loss of data.
	Warning	Risk of severe personal injury.
	New	This command or section is new for this release.

Table 2: Text Conventions

Convention	Description
Screen displays	This typeface indicates command syntax, or represents information as it appears on the screen.
The words enter and type	When you see the word “enter” in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says “type.”
[Key] names	Key names are written with brackets, such as [Return] or [Esc]. If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press [Ctrl]+[Alt]+[Del]
Words in <i>italicized type</i>	Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles.

Terminology

When features, functionality, or operation is specific to a switch family, the family name is used. Explanations about features and operations that are the same across all product families simply refer to the product as the “switch.”

Providing Feedback to Us

We are always striving to improve our documentation and help you work better, so we want to hear from you! We welcome all feedback but especially want to know about:

- Content errors or confusing or conflicting information.
- Ideas for improvements to our documentation so you can find the information you need faster.
- Broken links or usability issues.

If you would like to provide feedback to the Extreme Networks Information Development team about this document, please contact us using our short [online feedback form](#). You can also email us directly at InternalInfoDev@extremenetworks.com.

Getting Help

If you require assistance, contact Extreme Networks Global Technical Assistance Center using one of the following methods:

Web	www.extremenetworks.com/support
Phone	1-800-872-8440 (toll-free in U.S. and Canada) or 1-603-952-5000 For the Extreme Networks support phone number in your country: www.extremenetworks.com/support/contact
Email	support@extremenetworks.com To expedite your message, enter the product name or model number in the subject line.

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number
- A description of the failure
- A description of any action(s) already taken to resolve the problem (for example, changing mode switches or rebooting the unit)
- The serial and revision numbers of all involved Extreme Networks products in the network
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load and frame size at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any previous Return Material Authorization (RMA) numbers

Related Publications

The Extreme Security product documentation listed below can be downloaded from <http://documentation.extremenetworks.com>.

Extreme Security Analytics Threat Protection

- *Extreme Networks Security API Reference Guide*
- *Extreme Networks Security Application Configuration Guide*
- *Extreme Networks Security Ariel Query Language Guide*
- *Extreme Networks Security DSM Configuration Guide*
- *Extreme Security DSM Configuration Guide Addendum*
- *Extreme Networks Security Hardware Guide*
- *Extreme Networks Security Installation Guide*
- *Extreme Networks Security Juniper NSM Plug-in User Guide*
- *Extreme Networks Security Log Manager Administration Guide*
- *Extreme Networks Security Log Sources User Guide*
- *Extreme Networks Security Managing Log Sources Guide*
- *Extreme Networks Security Offboard Storage Guide*
- *Extreme Security Release Notes*

- *Extreme Networks Security Risk Manager Adapter Configuration Guide*
- *Extreme Networks Security Risk Manager Getting Started Guide*
- *Extreme Networks Security Risk Manager Installation Guide*
- *Extreme Networks Security Risk Manager Migration Guide*
- *Extreme Networks Security Risk Manager User Guide*
- *Extreme Networks Security Troubleshooting System Notifications Guide*
- *Extreme Networks Security Upgrade Guide*
- *Extreme Networks Security Vulnerability Manager Release Notes*
- *Extreme Networks Security Vulnerability Manager User Guide*
- *Extreme Networks Security WinCollect User Guide*
- *Extreme Networks SIEM Administration Guide*
- *Extreme Networks SIEM Getting Started Guide*
- *Extreme Networks SIEM High Availability Guide*
- *Extreme Networks SIEM Troubleshooting Guide*
- *Extreme Networks SIEM Tuning Guide*
- *Extreme Networks SIEM Users Guide*
- *Migrating Extreme Security Log Manager to Extreme SIEM*

Extreme Security Threat Protection

- *Extreme Security Intrusion Prevention System Hardware Replacement Guide*
- *Extreme Security Threat Protection Release Notes*

1 Ariel Query Language (AQL)

Ariel Query Language (AQL) deprecated versions

AQL functions

Logical and comparative operators

Event, flow and simarc fields for AQL queries

SELECT statement

WHERE clause

GROUP BY clause

ORDER BY clause

LIKE clause

COUNT function

The Ariel Query Language (AQL) is a structured query language that you use to communicate with the Ariel databases. Use AQL to manage event and flow data from the Ariel database.

Ariel Query Language (AQL) deprecated versions

Ariel Query Language (AQL) v1 and v2 are deprecated.

The command-line script, `/opt/gradar/bin/arielClient` is deprecated. The following warning message is displayed both before and after the results are returned:

```
WARNING: AQL v1 and v2 will be deprecated in the future. For information about using AQL v3, see the product documentation.
```

During your migration to v3, you can suppress the warning message by typing: `/opt/gradar/bin/arielClient | grep -v WARNING`

The Python client and the Advanced search option use AQL v3.

AQL fields changed in AQL V3

Ariel Query Language (AQL) V2 is deprecated in Extreme Security V7.2.4 and later. Some Ariel database fields were changed or removed in AQL V3. If you have queries that use these fields, you must replace them.

This table shows the new Ariel database fields.

Table 3: Fields that were replaced in AQL V3

Field name (AQL V2)	Replacement function name (AQL V3)
destinationAssetName	AssetHostname
deviceGroup	LogSourceGroupName
sourceAssetName	AssetHostname
eventDescription	QidName
destinationNetwork	NetworkName
endDate	DateFormat
endDateFormatted	DateFormat
eventProcessor	Processorname
identityUsername	AssetUser
identityMAC	AssetProperty
identityHostName	AssetHostname
identityNetBiosName	AssetHostname
identityGroupName	AssetProperty
identityExtendedField	AssetProperty
deviceDate	DateFormat
payloadHex	UTF8
protocol	ProtocolName
sourceNetwork	NetworkName
startDate	DateFormat
startDateFormatted	DateFormat
destinationAssetName	AssetHostname
sourceAssetName	AssetHostname
destinationNetwork	NetworkName
sourceNetwork	NetworkName
application	ApplicationName
destinationPayloadHex	UTF8
firstPacketDate	DateFormat
eventProcessorId	ProcessorName

This lists shows the Ariel database fields that were removed.

- partialorMatchList
- qidNumber
- token
- destinationHost

- destinationIPSearch
- destinationPortNA
- sourceHost
- sourceIPSearch
- sourcePortNA
- destinationDscpOnly
- anyDestinationFlag
- smallDestinationPayload
- smallDestinationPayloadHex
- destinationPrecedanceOnly
- lastPacketDate
- localHost
- remoteHost
- sourceDscpOnly
- anySourceFlag
- sourcePayloadHex
- smallSourcePayload
- smallSourcePayloadHex
- sourcePrecedanceOnly
- sourceHostString
- destinationHostString
- destinationNetwork
- application
- sourceNetwork
- smallPayload
- smallPayloadHex
- quickSearchMatches
- bitsPerSecond
- srcBitsPerSecond
- dstBitsPerSecond
- bytesPerSecond
- bytesPerPacket
- srcBytesPerPacket
- dstBytesPerPacket
- destinationByteRatio
- destinationPacketRatio
- packetsPerSecond
- sourceByteRatio
- sourcePacketRatio
- totalBytes
- totalPackets
- retentionBucket

- properLastPacketTime
- properLastPacketDate

AQL functions

Use Ariel Query Language (AQL) built-in functions to do calculations on data in the Ariel database.

Table 4: Basic functions

Operator	Description	Example
STR	Converts any parameter to a string.	STR(sourceIP)
STRLEN	Returns the length of this string.	STRLEN.(userName)
SUBSTRING	Copies a range of characters into a new string.	SUBSTRING(userName, 0, 3)
CONCAT	Concatenates all passed strings into 1 string.	CONCAT(userName, STR(sourceIP))
PARSEDATETIME	Returns the current time, which is expressed as milliseconds since the time 00:00:00 Coordinated Universal Time (UTC) on January 1, 1970.	PARSEDATETIME('1 week ago')
DATEFORMAT	Formats a time, which is expressed as milliseconds since the time 00:00:00 Coordinated Universal Time (UTC) on January 1, 1970 to a user-readable form.	DATEFORMAT(startTime, 'YYYY-MM-DD HH:mm:ss') as StartTime
NOW	Returns the current time that is expressed as milliseconds since the time 00:00:00 Coordinated Universal Time (UTC) on January 1, 1970.	NOW()
UTF8	Returns the UTF8 string of a byte array.	UTF8(payload)

Table 5: Aggregate functions

Operator	Information	Example
GROUP BY	Creates an aggregate on one or more columns.	SELECT sourceIP, COUNT(*) from events group by sourceIP, destinationIP
COUNT	Returns the count of the rows in the aggregate.	SELECT sourceIP, COUNT(*) from events group by sourceIP
UNIQUECOUNT	Returns the unique count of the value in the aggregate.	SELECT sourceIP, UNIQUECOUNT(category) from events group by sourceIP
FIRST	Returns the first entry of the rows in the aggregate.	SELECT sourceIP, FIRST(magnitude) from events group by sourceIP
SUM	Returns the sum of the rows in the aggregate.	SELECT sourceIP, SUM(sourceBytes) from flows group by sourceIP
AVG	Returns the average value of the rows in the aggregate.	SELECT sourceIP, AVG(magnitude) from events group by sourceIP

Table 5: Aggregate functions (continued)

Operator	Information	Example
MIN	Returns the minimum value of the rows in the aggregate.	<code>SELECT sourceIP, MIN(magnitude) from events group by sourceIP</code>
MAX	Returns the maximum value of the rows in the aggregate.	<code>SELECT sourceIP, MAX(magnitude) from events group by sourceIP</code>
HAVING	Allows operators on the result of a grouped by column.	<code>SELECT sourceIP, MAX(magnitude) as MAG from events group by sourceIP HAVING MAG > 5</code>

Table 6: External functions

Name	Description	Argument type	Description
HostName	Looks up a log source ID or a flow source ID.	NUMERIC	Log source ID or the flow source ID.
AssetHostname	Looks up a host name of an asset at a point in time.	VARCHAR DOUBLE	IP address, Time stamp Optional: If not specified, uses NOW ()
AssetProperty	Looks up a property for an asset at the current time.	VARCHAR OTHER DOUBLE	IP address, Property name, Time stamp Optional: If not specified, uses NOW ()
AssetUser	Looks up a user for an asset at a point in time.	VARCHAR DOUBLE	IP address, Timestamp Optional: If not specified, uses NOW ()
MatchesAsset Search	If the asset is contained in the results of the asset saved search it returns true.	VARCHAR VARCHAR	IP address, Saved Search Name
ReferenceMap	Looks up the value for a key in a reference map.	JAVA_OBJECT JAVA_OBJECT	String, String Example ReferenceMap ('IPLookup', 'userName')
ReferenceTable	Looks up the value for a column key in a table that is identified by a table key in a specific reference table collection.	VARCHAR JAVA_OBJECT JAVA_OBJECT	String, String, String (or IP address) Example ReferenceTable ('testTable', 'numKey', '100.10.10.1') or ReferenceTable ('testTable', 'numKey', sourceIP)

Table 6: External functions (continued)

Name	Description	Argument type	Description
Reference MapSet Contains	If a value is contained in a reference set that is identified by a key in a specific reference map of set it returns <code>true</code> .	VARCHAR JAVA_OBJECT JAVA_OBJECT	String, String, String Example ReferenceMap SetContains('RiskyUsersForIps', 'sourceIP', 'userName')
ReferenceSet Contains	If a value is contained in a specific reference set, it returns <code>true</code> .	VARCHAR JAVA_OBJECT	String, String Example ReferenceSetContains ('MySet', 'SourceIP')
CategoryName	Looks up the name of a category by its ID.	NUMERIC	Category ID
LogSource Group Name	Looks up the name of a log source group by its log source group ID.	NUMERIC	Device group list Example LogSourceGroupName(deviceGroupList)
QidDescription	Looks up the description of a QID by its QID.	NUMERIC	QID
QidName	Looks up the name of a QID by its QID.	NUMERIC	QID
Application Name	Returns the name of a flow application.	NUMERIC	Application ID
LogSource Name	Looks up the name of a log source by its log source ID.	NUMERIC	Log source ID Example LogSourceName(logSourceId)
LogSource Type Name	Looks up the name of a log source type by its device type.	Types . NUMERIC	Device type Example LogSourceTypeName(deviceType)
UTF-8	Returns the UTF-8string.	VARBINARY	A byte array Example Payload
StrLen	Returns the length of this string.	VARCHAR	String

Table 6: External functions (continued)

Name	Description	Argument type	Description
<code>Str</code>	Converts parameter to string.	<code>JAVA_OBJECT</code>	String
<code>SubString</code>	Copies a range of characters into a new string.	<code>VARCHAR</code> <code>NUMERIC</code> <code>NUMERIC</code>	A String, a start that is offset, and a length
<code>Concat</code>	Concatenates all passed strings into 1 string.	<code>VARCHARNUMERIC</code> <code>NUMERIC</code>	List of strings
<code>ParseDate time</code>	Returns the current time, which is expressed as milliseconds since the time 00:00:00 Coordinated Universal Time (UTC) on January 1, 2014.	<code>VARCHAR</code>	A String that represents a date and time
<code>Now</code>	Returns the current time, which is expressed as milliseconds since the time 00:00:00 Coordinated Universal Time (UTC) on January 1, 2014.	<code>NULL</code>	None
<code>ProtocolName</code>	Returns the name of a protocol, which is based on a protocol ID number.	<code>NUMERIC</code>	Protocol ID number
<code>InOffense</code>	If an event or flow belongs to the specified offense, it returns <code>true</code> .	<code>NUMERIC</code>	Offense ID Example <pre>SELECT * FROM events WHERE InOffense(123)SELECT * FROM flows WHERE InOffense(123)</pre>
<code>InCIDR</code>	If the IP/column, specified is contained in, or equal to, the specified IP/CIDR, it returns <code>true</code> .	<code>VARCHAR</code> , <code>OTHER</code>	IP/CIDR, IP address Example <pre>...WHERE InCIDR('172.16.0.0/16', sourceip) AND ...</pre>
<code>NetworkName</code>	Looks up the network name from the network hierarchy for the <code>Host</code> that is passed in.	<code>OTHER</code>	Host property Example <pre>NetworkName(sourceip)</pre>

Table 6: External functions (continued)

Name	Description	Argument type	Description
RuleName	Returns one or more rule names that are based on the rule ID or IDs that are passed in.	INTEGER	A single rule ID, or a list of rule IDs. Example RuleName(creEventList), RuleName(1033)
Long	Parses a string that represents a number into a Long (integer) data type.	VARCHAR	A string that represents a number. Example Long('1234')
Double	Parses a string that represents a number into a Double (integer) data type.	VARCHAR	A string that represents a number. Example Double('1234')
DomainName	Looks up the domain name based on domain ID.	NUMERIC	domain ID Example DomainName(domainID)

Logical and comparative operators

Logical operators are used in AQL statements to determine any equality or difference between values. By using logical operators in the WHERE clause of an AQL statement, the results returned are restricted/filtered to those that match the conditions in the WHERE clause. The following table lists the supported operators.

Table 7: Operators for the Ariel API

Operator	Information	Example
=	Compares 2 values and returns true if they are equal.	...WHERE sourceIP = destinationIP
!=	Compares 2 values and returns true if they are not equal.	...WHERE sourceIP != destinationIP)
(and)	Use brackets to nest components of a WHERE or HAVING clause to create complex Boolean expressions.	...WHERE (sourceIP = destinationIP) AND (sourcePort = destinationPort)
< and <=	Compares two values and returns true if the left value is less than or, less than or equal to, the right value.	...WHERE sourceBytes < 64 and destinationBytes <= 64
> and >=	Compares two values and returns true if the left value is greater than or, greater than or equal to, the right value.	...WHERE sourceBytes > 64 and destinationBytes >= 64

Table 7: Operators for the Ariel API (continued)

Operator	Information	Example
*	Multiplies 2 values and returns the result.	<code>...WHERE sourceBytes * 1024 < 1</code>
/	Divides 2 values and returns the result.	<code>...WHERE sourceBytes / 8 > 64</code>
+	Adds 2 values and returns the result.	<code>...WHERE sourceBytes + destinationBytes < 64</code>
-	Subtracts 1 value from another and returns the result.	<code>...WHERE sourceBytes - destinationBytes > 0</code>
^	Takes a value and raises it to the specified power and returns the result.	<code>...WHERE sourceBytes ^ 2 < 256</code>
%	Takes the modulo of a value and returns the result.	<code>...WHERE sourceBytes % 8 == 7</code>
AND	Takes the left side of a statement and the right side of a statement and returns true if both are true.	<code>...WHERE (sourceIP = destinationIP) AND (sourcePort = destinationPort)</code>
OR	Takes the left side of a statement and the right side of a statement and returns true if either one is true.	<code>...WHERE (sourceIP = destinationIP) OR (sourcePort = destinationPort)</code>
NOT	Takes in a statement and returns true if the statement evaluates to false.	<code>...WHERE NOT (sourceIP = destinationIP)</code>
IS NULL	Takes in a value and returns true if the value is null.	<code>...WHERE userName IS NULL</code>
NOT NULL	Takes in a value and returns true if the value is not null.	<code>...WHERE userName IS NOT NULL</code>
BETWEEN (X , Y)	Takes in a left side and two values and returns true if the left side is between the two values.	<code>...WHERE magnitude BETWEEN 1 AND 5</code>
LIMIT	Limits the number of results to the provided number.	<code>...WHERE magnitude > 5 LIMIT 10</code>
ORDER BY (ASC , DESC)	Orders the result set by the provided columns.	<code>SELECT * FROM EVENTS ORDER BY sourceIP DESC</code>
COLLATE	Parameter to order by that allows a BCP47 language tag to collate.	<code>SELECT * FROM EVENTS ORDER BY sourceIP DESC COLLATE 'de-CH'</code>
INTO	Creates a named cursor that contains results that can be queried at a different time.	<code>SELECT * FROM EVENTS INTO 'MyCursor' WHERE....</code>

Table 7: Operators for the Ariel API (continued)

Operator	Information	Example
START	You can pass a time interval to start selecting data from in the format <code>yyyy-MM-dd HH:mm</code> . Use in combination with STOP.	<code>...WHERE userName IS NULL START '2014-01-01 12:00' STOP '2014-02-01 17:00'</code>
STOP	You can pass a time interval to stop selecting data from in the format <code>yyyy-MM-dd HH:mm</code> . Use in combination with START.	<code>...WHERE userName IS NULL START '2014-01-01 12:00' STOP '2014-02-01 17:00'</code>
LAST	You can pass a time interval to select data from. Valid intervals are MINUTES, HOURS, and DAYS	<code>...WHERE userName IS NULL LAST 6 HOURS</code>
LIKE	Matches if the string passed, is LIKE the passed value. % is a wildcard.	<code>...WHERE userName LIKE '%bob%'</code>
ILIKE	Matches if the string passed, is LIKE the passed value in a case-insensitive manner. % is a wildcard.	<code>...WHERE userName ILIKE '%bob%'</code>
MATCHES	Matches if the string matches the provided regular expression.	<code>...WHERE userName MATCHES '^.bob.\$'</code>
IMATCHES	Matches if the string matches the provided regular expression in a case-insensitive manner.	<code>...WHERE userName IMATCHES '^.bob.\$'</code>
TEXT SEARCH	Full-text search for the passed value. You can also do full-text searches by using the Quick filter in the Extreme Security user interface. For information about Quick filter functions, see the <i>Extreme Networks SIEM Users Guide</i> . TEXT SEARCH is valid with AND operators. You can't use TEXT SEARCH with OR or other operators; otherwise you will get a syntax error.	<code>...WHERE TEXT SEARCH 'firewall' AND ...</code>

Examples of logical and comparative operators

- To sort events that are unparsed, type the following query: `SELECT * FROM events WHERE payload = 'false'`
- To sort flows to find a specific source IP address that has an offense, type the following query: `SELECT * FROM events WHERE sourceIP = '231.12.37.17' AND hasOffense = 'true'`
- You can do a **Quick filter** search in AQL. To sort events for "firewall", type the following query: `SELECT QIDNAME(qid) AS EventName, * from events where TEXT SEARCH 'firewall'`

Event, flow and simarc fields for AQL queries

Use the Ariel Query Language (AQL) to retrieve specific fields from the events, flows and simarc table in the Ariel database.

Supported flow fields for AQL queries

The flow fields that you can query are listed in the following table.

Table 8: Supported flow fields for AQL queries

Field name	Description
applicationId	Application ID
category	Category
credibility	Credibility
destinationASN	Destination ASN
destinationBytes	Destination bytes
destinationDSCP	Destination DSCP
destinationFlags	Destination flags
destinationIP	Destination IP
destinationIfIndex	Destination if index
destinationPackets	Destination packets
destinationPayload	Destination payload
destinationPort	Destination port
destinationPrecedence	Destination precedence
destinationTOS	Destination QoS
destinationv6	IPv6 destination
processorID	Event processor ID
fullMatchList	Full match list
firstPacketTime	First packet time
flowBias	Flow bias
flowDirection	Flow direction <ul style="list-style-type: none"> • local-to-local (L2L) • local-to-remote (L2R) • remote-to-local (R2L) • remote-to-remote (R2R)
flowInterfaceID	Flow interface ID
flowSource	Flow Source
flowType	Flow type
geographic	Matches geographic location
hasDestinationPayload	Has destination payload

Table 8: Supported flow fields for AQL queries (continued)

Field name	Description
hasOffense	Has offense payload
hasSourcePayload	Has source payload
icmpCode	Icmp code
icmpType	ICMP type or code
flowInterface	Flow interface
intervalId	Interval ID
isDuplicate	Duplicate event
lastPacketTime	Last packet time
partialMatchList	Partial match list
protocol	Protocol
protocolId	Protocol ID
qid	Qid
relevance	Relevance
retentionBucket	Retention bucket dummy
severity	Severity
sourceASN	Source ASN
sourceBytes	Source bytes
sourceDSCP	Source DSCP
sourceFlags	Source flags
sourceIP	Source IP
sourceIfIndex	Source if index
sourcePackets	Source packets
sourcePayload	Source payload
sourcePort	Source port
sourcePrecedence	Source precedence
sourcev6	IPv6 source
startTime	Start time
viewObjectPair	View object pair

Supported event fields for AQL queries

The event fields that you can query are listed in the following table.

Table 9: Supported event fields for AQL queries


Field name	Description
category	Low-level category
creEventList	Matched custom rule
credibility	Credibility
destinationMAC	Destination MAC
destinationPort	Destination port
destinationv6	IPv6 destination
deviceTime	Log source time
deviceType	Log source type
domainID	Domain ID
	 Note Log Manager only
duration	Duration
endTime	End time
eventCount	Event count
eventDirection	Event direction: <ul style="list-style-type: none"> • local-to-Local (L2L) • local-to-remote (L2R) • remote-to-local (R2L) • remote-to-remote (R2R)
processorId	Event Processor ID
hasIdentity	Has identity
hasOffense	Associated with offense
highLevelCategory	High-level category
isCREEvent	Is custom rule event
magnitude	Magnitude
payload	Payload
postNatDestinationIP	Destination IP after NAT
postNatDestinationPort	Destination port after NAT
postNatSourceIP	Source IP after NAT
postNatSourcePort	Source port after NAT
preNatDestinationIP	Destination IP before NAT
preNatDestinationPort	Destination port before NAT
preNatSourceIP	Source IP before NAT
preNatSourcePort	Source port before NAT

Table 9: Supported event fields for AQL queries (continued)

Field name	Description
protocolID	Protocol
qid	Event name ID
relevance	Relevance
severity	Severity
sourceIP	Source IP
sourceMAC	Source MAC
sourcePort	Source port
sourcev6	IPv6 source
startTime	Start time
isunparsed	Event is unparsed
userName	User name

Supported simarc fields for AQL queries

The simarc fields that you can query are listed in the following table.

Table 10: Supported simarc fields for AQL queries

Field name	Description
destinationPort	Destination port key creator
destinationType	Destination type key creator
deviceId	Device key creator
direction	Direction key creator
eventCount	Event count key creator
eventFlag	Flag key creator
applicationId	Application ID key creator
flowCount	Flow count key creator
destinationBytes	Destination bytes key creator
flowSource	Flow source key creator
sourceBytes	Source bytes key creator
lastPacketTime	Time key creator
protocolId	Protocol key creator
source	Source key creator
sourceType	Source type key creator
sourceRemoteNetwork	Source remote network key creator
destinationRemoteNetwork	Destination remote network key creator

Table 10: Supported simarc fields for AQL queries (continued)

Field name	Description
sourceCountry	Source geographic key creator
destinationCountry	Destination geographic key creator
destination	Destination key creator
creEventList	Normalized event properties CRE event list
partialMatchList	Normalized event properties partial match list

SELECT statement

Use the SELECT statement to retrieve specific data from the events or flows table in the Ariel database. A SELECT operation is called a *query*.

Syntax

```
SELECT selectList
    FROM joinClauses
    [WHERE searchCondition]
    [GROUP BY groupClause]
    [ORDER BY orderClause]
```

Usage

A SELECT statement can include one or more fields from the flow or event tables. Use an asterisk, *, to denote all columns. All field names are case-sensitive. However, SELECT and FROM statements are not case-sensitive.

Overriding the time settings passed to the AQL query

The SELECT statement supports an **arielttime** option, which overrides the time settings.

You can limit the time period for which an AQL query is evaluated.

You can use the START and STOP keywords.

Example

```
SELECT sourceIP FROM events START '2014-05-02 09:25' stop '2014-05-02
09:30'
```

You can also use the LAST keyword.

Example

```
SELECT * FROM events LAST 15 MINUTES
SELECT * FROM events LAST 1 HOURS
SELECT * FROM events LAST 2 DAYS
```

Examples of SELECT statements that use CIDR ranges

You can also use SELECT statements for CIDR-based queries. To query by source IP address, `sourceIP`, or by destination IP address, `destinationIP`, use the following format:

```
SELECT <query item> FROM <flows/events> WHERE
<sourceCIDR/destinationCIDR> = '<CIDR Range>'
```

Example

```
SELECT * FROM flows WHERE sourceCIDR = '10.100.100/24'
```

To return all flows that are coming from the 10.100.100 subnet or capture flows that are coming from and into the subnet, use the regular OR expression.

Example

```
SELECT * FROM flows WHERE sourceCIDR = '10.100.100/24' OR
destinationCIDR = '10.100.100/24'
```

To query when `source IP` is contained in the 192.168.222.0/24 range, use the following format:

```
SELECT <query item> FROM <events> WHERE
<INCIDR> = '<INCIDR Range>'
```

Example

```
SELECT * FROM events WHERE INCIDR('192.168.222.0/24', sourceIP)
```

To query when `source IP` is not contained in the 192.168.222.0/24 range, use the following format:

```
SELECT <query item> FROM <events> WHERE
<INCIDR> != '<INCIDR Range>'
```

Example

```
SELECT * FROM events WHERE NOT INCIDR('192.168.222.0/24', sourceIP)
```

WHERE clause

Restrict your AQL queries by using WHERE clauses. The WHERE clause describes the filter criteria to apply to the query and filters the resulting view to accept only those events or flows that meet the specified condition.

Syntax

```
WHERE searchCondition
```

A *searchCondition* is a combination of logical and comparison operators that together make a test. Only those input rows that pass the test are included in the result.

Examples of WHERE clauses

The following query example shows events that have a severity level of greater than 9 are selected from a category.

```
SELECT sourceIP, category, credibility FROM events WHERE
severity > 9 AND category = 5013
```

You can change the order of evaluation by using parentheses. The search conditions that are enclosed in parentheses are evaluated first.

```
SELECT sourceIP, category, credibility FROM events WHERE
(severity > 9 AND category = 5013) OR (severity < 5 and
credibility > 8)
```

GROUP BY clause

Use the GROUP BY clause to aggregate your data. To provide meaningful results of the aggregation, usually, data aggregation is combined with arithmetic functions on remaining columns.

Syntax

```
GROUP BY groupClause
```

You can use aggregate functions in Ariel Query Language (AQL) queries to summarize information from multiple rows. The aggregate functions that are supported are shown in the following table.

Table 11: Aggregate functions

Function	Description
GROUP BY	Creates an aggregate on one or more columns.
COUNT	Returns the count of the rows in the aggregate.
UNIQUECOUNT	Returns the unique count of the value in the aggregate.
FIRST	Returns the first entry of the rows in the aggregate.
SUM	When used with numeric data, returns the sum of the values. When used with categorical data, it returns the union of the categorical values.
AVG	Returns the average value of the rows in the aggregate.
MIN(expr)	Returns the lowest value of the rows in the aggregate..

Table 11: Aggregate functions (continued)

Function	Description
MAX(expr)	Returns the highest value of the rows in the aggregate.
HAVING	Allows operators on the result of a grouped by column.

Examples of GROUP BY clauses

The following query example shows IP addresses that sent more than 1 million bytes within all flows in a specific time.

```
select sourceIP, SUM(sourceBytes) from flows where sourceBytes >
1000000 group by sourceIP
```

The results might look similar to the following output.

```
-----
| sourceIP | SUM_sourceBytes |
-----
| 64.124.201.151 | 4282590.0 |
| 10.105.2.10 | 4902509.0 |
| 10.103.70.243 | 2802715.0 |
| 10.103.77.143 | 3313370.0 |
| 10.105.32.29 | 2467183.0 |
| 10.105.96.148 | 8325356.0 |
| 10.103.73.206 | 1629768.0 |
-----
```

However, if you compare this information to a non-aggregated query, the output displays all the IP addresses that are unique, as shown in the following output:

```
-----
| sourceIP | sourceBytes |
-----
| 64.124.201.151 | 1448629 |
| 10.105.2.10 | 2412426 |
| 10.103.70.243 | 1793095 |
| 10.103.77.143 | 1449148 |
| 10.105.32.29 | 1097523 |
| 10.105.96.148 | 4096834 |
| 64.124.201.151 | 2833961 |
| 10.105.2.10 | 2490083 |
| 10.103.73.206 | 1629768 |
| 10.103.70.243 | 1009620 |
| 10.105.32.29 | 1369660 |
| 10.103.77.143 | 1864222 |
| 10.105.96.148 | 4228522 |
-----
```

To view the maximum number of events, use the following syntax:

```
SELECT MAX(eventCount) FROM events
```

To view the number of average events from a source IP, use the following syntax:

```
SELECT AVG(eventCount) FROM events GROUP BY sourceIP
```

The output displays the following results:

```
-----
| sourceIP | protocol |
-----
| 64.124.201.151 | TCP.tcp.ip |
| 10.105.2.10 | UDP.udp.ip |
| 10.103.70.243 | UDP.udp.ip |
| 10.103.77.143 | UDP.udp.ip |
| 10.105.32.29 | TCP.tcp.ip |
| 10.105.96.148 | TCP.tcp.ip |
| 64.124.201.151 | TCP.tcp.ip |
| 10.105.2.10 | ICMP.icmp.ip |
-----
```

ORDER BY clause

Use the ORDER BY clause to sort the resulting view that is based on expression results. The order is sorted by ascending or descending sequence.

Syntax

```
ORDER BY orderClause
```

Only one field can be used in the ORDER BY clause. You can switch sorting between ascending or descending by appending the ASC or DESC keyword to the order by clause.

Combining GROUP BY and ORDER BY clauses to create data

To determine the top abnormal events or the most bandwidth-intensive IP addresses, you can combine GROUP BY and ORDER BY clauses in a single query. When you combine the clauses, you create data, such as TopN lists. For example, the following query displays the most traffic intensive IP address in descending order:

```
SELECT sourceIP, SUM(sourceBytes) FROM flows GROUP sourceIP
ORDER BY SUM(sourceBytes) DESC
```

Examples of ORDER BY clauses

To query AQL to return results in descending order. use the following syntax:

```
SELECT sourceBytes, sourceIP FROM flows WHERE sourceBytes >
1000000 ORDER BY sourceBytes
```

To display results in ascending order, use the following syntax:

```
SELECT sourceBytes, sourceIP FROM flows WHERE sourceBytes >
1000000 ORDER BY sourceBytes ASC
```

LIKE clause

Use the LIKE clause to retrieve partial string matches in the Ariel database.

Syntax

```
ORDER BY orderClause
```

You can search fields by using the LIKE clause.

The following wildcard options are supported by the Ariel Query Language (AQL):

Table 12: Supported wildcard options for LIKE clauses

Wildcard character	Description
%	Matches a string of zero or more characters
_	Matches any single character

Examples of LIKE clauses

To match names such as Joe, Joanne, Joseph, or any other name that begins with Jo, type the following query:

```
SELECT * FROM events WHERE userName LIKE 'jo%'
```

To match names beginning with Jo that are three characters long, such as, Joe or Jon, type the following query:

```
SELECT * FROM events WHERE userName LIKE 'Jo_'
```

You can enter the wildcard option at any point in the command, as shown in the following examples.

```
SELECT * FROM flows WHERE sourcePayload LIKE '%xyz'
SELECT * FROM events WHERE payload LIKE '%xyz%'
SELECT * FROM events WHERE payload LIKE '_yz'
```

Examples of string matching keywords

The keywords, ILIKE and IMATCHES are case-insensitive versions of LIKE and MATCHES.

```
SELECT qidname(qid) as test FROM events WHERE test LIKE 'Information%'
SELECT qidname(qid) as test FROM events WHERE test ILIKE 'inForMatiOn%'
```

```
SELECT qidname(qid) as test FROM events WHERE test MATCHES '.*Information.*'  
SELECT qidname(qid) as test FROM events WHERE test IMATCHES '.*Information.*'
```

COUNT function

The COUNT function returns the number of rows that satisfy the WHERE clause of a SELECT statement.

If the SELECT statement does not have a WHERE clause, the COUNT function returns the total number of rows in the table.

Syntax

```
COUNT
```

Examples

To count all events with credibility equal to or greater than 9, type the following query, type the following query:

```
SELECT COUNT() FROM events WHERE credibility >= 9
```

Index

A

AQL 8
Ariel Query Language 8
ariertime option 23, 24

C

command-line options
 COUNT function 28
 GROUP BY 24, 25
 LIKE clause 27
 ORDER BY clause 26
 SELECT clause 22, 23
 WHERE clause 23, 24
comparative operators
 WHERE clause 23, 24
conventions, guide
 notice icons 4
 text 5
COUNT function
 description 28
customer support
 contact information 4

E

events and flows
 field list 18, 19, 21

F

functions
 fields 8
 supported list 11

G

GROUP BY
 description 24, 25

L

LIKE clause
 description 27
logical operators
 WHERE clause 23, 24

N

network administrator
 description 4

O

ORDER BY clause
 description 26

overriding time settings 23, 24

S

SELECT clause
 description 22, 23

T

time settings 23, 24

W

WHERE clause
 description 23, 24