

# User Guide: Observability Analytics Query Builder (Private Preview)

Private preview sign up form: <https://forms.gle/76VBUXpP9xBzXNcN8>

## Table of content

[Table of content](#)

[Introduction](#)

[Enable Observability Analytics](#)

[Observability UI Layout](#)

[Query Builder UI Component](#)

[Sample Use Cases](#)

[Example 1: Search '404' in all fields](#)

[Example 2: Search 'stderr' in log\\_name field](#)

[Example 3: Select fields with filters \(equality filter\)](#)

[Example 4: Alias selected fields, with regex filters](#)

[Example 5: Alias selected field with regex extraction](#)

[Example 6: Fields with aggregation and grouping, filter with IS NOT NULL](#)

[Example 7: Group by hour truncation from timestamp](#)

[Example 8: Filter and some number of fields with sort](#)

[Example 9: Some number of fields with groupings and a COUNT aggregation](#)

[Example 10: Filter and some number of fields with aggregations including aliases, regexp, COUNT, grouping, sort, and search term](#)

[Example 11: Regex extraction - the top 20 files accessed](#)

[Supported Comparison Operators](#)

[Not Supported Features](#)

[Coming Soon](#)

[To be scheduled](#)

[FAQ](#)

## Introduction

We are excited to announce that Observability Analytics (OA) query builder is now available for private preview!

If you are a Cloud Ops's [Observability Analytics](#) user (also known as Log Analytics), this feature is for you!

Observability query builder providing below key benefits:

- **No more writing SQL query:** Generating queries and getting results through a guided UI, no longer need to write SQL for most of your analytics questions. And you can continue writing SQL if you like.
- **Easy to extract, cast, and transform JSON data:** Query builder is designed to streamline some of the complexities of working with Observability Analytics data, such as extracting, casting, and transforming JSON data.
- **Toggle between builder and SQL:** You can toggle between query builder and automatically generated SQL, and make edits with the SQL to achieve more complex query.
- **Work with Log Scope:** Query builder enables you to query with log scope. So you can keep the scope consistent when doing deeper analysis and troubleshooting in log explorer.

This user guide will cover steps for enabling the query builder and provide some examples for common use cases. Its target users who are in the private preview. More detailed reference documentation will be available closer to public preview release.

## Enable Observability Analytics

In order to use the query builder, first [enable Log Analytics](#).

Then [Sign up](#) users from your organization using their email address in the private preview form.

Once users are enabled, they will be notified by the google contact person.

## Observability UI Layout

**Log view, trace views, analytics views, metrics view viewer:** show the list of available views

**Log Schema viewer:** Display log schema, including nested fields

**Query Building area:**

- **View select:** Selecting log view, **log scope**, metrics view, trace view and analytics views,
- Construct query and answer questions by selecting fields, filters, aggregations, sortings, apply regex, alias, casting value type (coming soon).

**Query result plane:**

- Display results from the construct.
- Data time range is automatically applied from the time range picker on the top right

The screenshot shows the Google Cloud Observability Analytics interface. On the left, there is a sidebar with a 'Views' section labeled 'Logviews viewer' and a 'Schema' section labeled 'Schema viewer'. The main area is titled 'Observability Analytics' and contains a 'Query' section labeled 'Query building area' and a 'Results (100)' section labeled 'Query result plane'. The 'Query building area' includes a 'View' dropdown, a search bar, and fields for 'log\_id', 'location', 'json\_message', and 'response\_size'. The 'Results (100)' section shows a table of results with columns 'log\_id', 'oa0\_', 'json\_message', and 'response\_size'. A red arrow points from the 'Query building area' to the 'Results (100)' section, indicating a toggle to see/edit generated SQL.

## Query Builder UI Component

The query builder UI is split into several fields.

This detailed view of the Query Builder UI shows the following components:

- View:** A dropdown menu showing '\_Default\_Default' and a search bar with the text '404'.
- Fields:** A row of buttons for 'log\_id', 'GROUP BY', 'severity', 'JSON\_message', and 'Add field'.
- Filters:** A row of buttons for 'location = us-east1', 'project\_id =~ /product/(lw+)\s', and 'Add filter'.
- Sorts:** A row of buttons for 'log\_id', a dropdown arrow, and 'Add sort'.
- Limit:** A button labeled 'Limit' followed by the number '100' and a trash icon.

- “View” - allows you to pick the base view for the query, which will be presented in the FROM section. In addition to supporting log views and analytics views, there is also support for [Log Scopes](#), which allow querying multiple views at once.
- “Search all fields” - Provides easy filtering using a simple text based search against the entire row, similar to the Logs Explorer’s search bar.
- “Fields” - Allows you to specify the columns that will appear in the output table

- “Filters” - Allows you to apply filters against fields of the source data
- “Sorts” - Allows you to specify the columns to sort by
- “Limit” - Set the number of output rows

## Sample Use Cases

### Example 1: Search '404' in all fields

This query builder construct allows you to search term '404' across all fields of the log, and return the log as long as there is a match in any field of the log.

Observability Analytics

< Last 10 minutes ADT

+

QueryRecent (7)Saved (59)

Save

Run query

View\_Default\_Default

404

<> SQL

Fields

Add field

Filters

Add filter

Sorts

Add sort

Limit10000

### Corresponding manual query

```
SQL
SELECT
*
FROM
query-builder-test-project.global._Default._Default
WHERE
SEARCH('query-builder-test-project.global._Default._Default', '404')
LIMIT
10000
```

### Expected results

Row	log_id	text_payload	timestamp
	STRING	STRING	TIMESTAMP
1	stdout	10.0.3.1 -- [02/May/2025:18:29:06 +0000] "GET /sendgrid.env HTTP/1.1" 404 153 "-" "I9explore/1.2.2" "-"	2025-05-02 14:29:06.268 EDT

## Example 2: Search 'stderr' in log\_name field

Observability Analytics

< Last 12 hours EDT >

Query Recent (6) Saved (54) Save Run query

View \_Default.\_Default Q stderr <> SQL

Fields log\_name X Add field

Filters Add filter

Sorts Add sort Limit 10000

### Corresponding manual query

```
SQL
SELECT
  log_name
FROM
  query-builder-test-project.global._Default._Default
WHERE
  SEARCH('query-builder-test-project.global._Default._Default', 'stderr')
LIMIT
  10000
```

### Expected results

Row	log_name
	STRING
1	projects/query-builder-test-project/logs/stderr
2	projects/query-builder-test-project/logs/stderr

## Example 3: Select fields with filters (equality filter)

This query builder construct allows you to select log name, resource type, resource location, JSON payload from logs, where resource type is k8s\_container and location is us-central1

View
\_Default.\_Default
Search all fields

Fields
log\_name
type
location
json\_payload
Add field

Filters
location = us-central1
type = k8s\_container
Add filter

Sorts
Add sort
Limit
100

Corresponding manual query

```
SQL
SELECT
  log_name,
  resource.type,
  JSON_VALUE( resource.labels.location ),
  json_payload
FROM
  query-builder-test-project.global._Default._Default
WHERE
  resource.type = 'k8s_container'
  AND JSON_VALUE( resource.labels.location ) = 'us-central1'
LIMIT
  100
```

Expected results

Row	log_name STRING	type STRING	oa0_ STRING	json_payload
1	projects/query-builder-test-pr oject/logs/stdout	k8s_container	us-central1	Null
2	projects/query-builder-test-pr oject/logs/stdout	k8s_container	us-central1	Null
3	projects/query-builder-test-pr oject/logs/stdout	k8s_container	us-central1	Null

Example 4: Alias selected fields, with regex filters

Selecting resource type, resource location, severity and log name where resource locations are in all of us-east locations.

View
\_Default.\_Default
Search all fields

Fields
resource\_type
resource\_location
severity
log\_name
Add field

Filters
location == us-east\d+
Add filter

Sorts
Add sort
Limit
100

Corresponding manual query

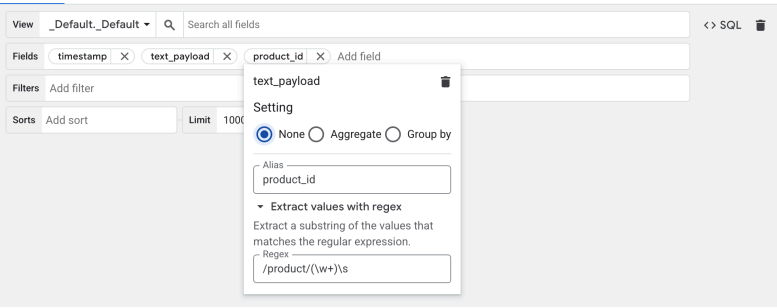
```
SQL
SELECT
  resource.type AS resource_type,
  JSON_VALUE( resource.labels.location ) AS resource_location,
  severity,
  log_name
FROM
  query-builder-test-project.global._Default._Default
WHERE
  REGEXP_CONTAINS( JSON_VALUE( resource.labels.location ), 'us-east\\d+')
LIMIT
  100
```

Expected results

Row	resource_type STRING	resource_location STRING	severity STRING	log_name STRING
1	k8s_cluster	us-east1-a	DEFAULT	projects/query-builder-test-project/ logs/clouddaudit.googleapis.com% 2Factivity
2	k8s_cluster	us-east4-b	DEFAULT	projects/query-builder-test-project/ logs/clouddaudit.googleapis.com% 2Factivity

Example 5: Alias selected field with regex extraction

Selecting timestamp, text\_payload, and an aliased product\_id from the text\_payload column with a regex extraction applied.



Corresponding manual query

```
SQL
SELECT
  timestamp,
  text_payload,
  REGEXP_EXTRACT( text_payload, '/product/(\\w+)\\s') AS product_id
FROM
  query-builder-test-project.global._Default._Default
```

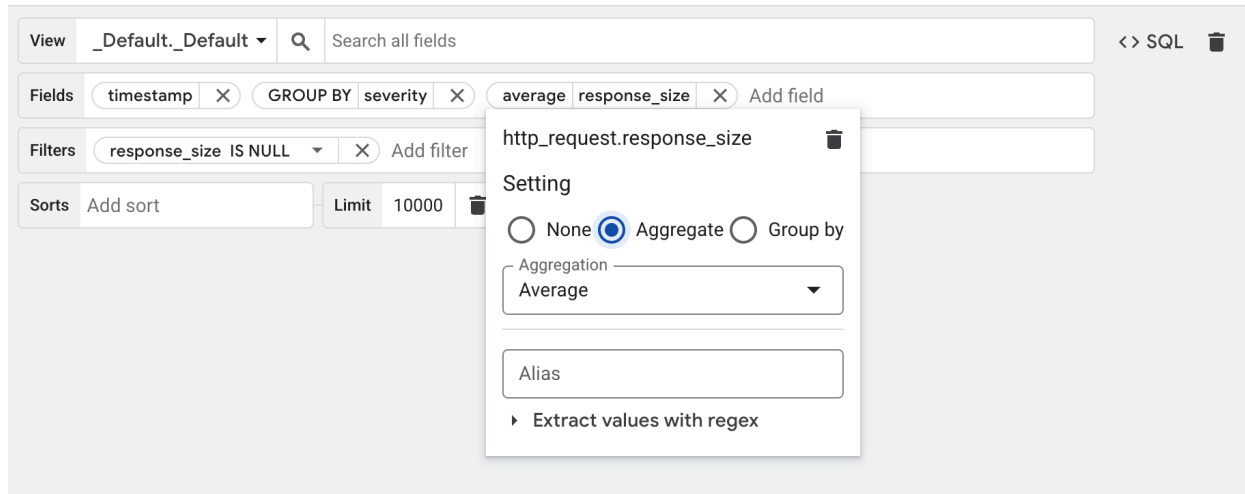
Expected results

Row	timestamp	text_payload	product_id
	TIMESTAMP	STRING	STRING
1	2025-05-02 10:16:29.749 EDT	GET /product/1PUX7V6EV0 12345 12345(100.00%)   11 2 3333 00   0.00 0.00	1PUX7V6EV0
2	2025-05-02 10:16:31.755 EDT	GET /product/1PUX7V6EV0 12345 85483(100.00%)   11 2 3333 00   0.00 0.00	1PUX7V6EV0
3	2025-05-02 10:22:44.673 EDT	GET /product/1PUX7V6EV0 12345 12345(100.00%)   11 2 3333 00   0.30 0.30	1PUX7V6EV0

Example 6: Fields with aggregation and grouping, filter with IS NOT NULL

Select average response size and group by severity and timestamp





## Corresponding manual query

```
SQL
SELECT
  timestamp,
  severity,
  AVG(proto_payload.request_log.response_size) AS response_size
FROM
  `query-builder-test-project.global._Default._Default`
WHERE
  proto_payload.request_log.response_size IS NOT NULL
GROUP BY
  timestamp,
  severity
LIMIT
  20;
```

## Expected results

Row	timestamp	severity	response_size
	TIMESTAMP	STRING	INTEGER
1	2025-05-02 14:00:00.000 EDT	INFO	431
2	2025-05-02 13:00:00.000 EDT	NOTICE	2540

## Example 7: Group by hour truncation from timestamp

Group response size by hours exacted from timestamp

View **\_Default.\_Default** Search all fields <> SQL

Fields **GROUP BY HOUR** **timestamp** **average** **response\_size** Add field

Filters Add filter

Sorts Add sort **Limit** 10000

View **\_Default.\_Default** Search all fields <> SQL

Fields **GROUP BY HOUR** **timestamp** **average** **response\_size** Add field

Filters

Sorts

**timestamp**

Setting

☐ None ☒ Group by

Truncation Granularity **Hour**

Alias

▶ Extract values with regex

## Corresponding manual query

```
SQL
SELECT
  TIMESTAMP_TRUNC(timestamp, HOUR) as event_hour,
  AVG(proto_payload.request_log.response_size) AS avg_response_size
FROM
  `query-builder-test-project.global._Default._Default`
WHERE
  proto_payload.request_log.response_size IS NOT NULL
GROUP BY
  event_hour
LIMIT
  20;
```

## Expected results

Row	event_hour TIMESTAMP	response_size FLOAT
1	2025-05-02 14:00:00.000 EDT	756.93939393939
2	2025-05-02 08:00:00.000 EDT	761.92473118279

## Example 8: Filter and some number of fields with sort

The screenshot shows a query builder interface with the following components:

- View:** A dropdown menu set to `_Default._Default` with a search icon and a search bar labeled "Search all fields".
- Fields:** A section containing two "GROUP BY" clauses: `response_size` and `severity`. Each clause has a close button (X). An "Add field" button is also present.
- Filters:** A section with an "Add filter" button.
- Sorts:** A section with a sort clause `response_size` and a downward arrow. A close button (X) and an "Add sort" button are also present. To the right, there is a "Limit" section set to `10000` with a trash icon.
- Sort Order Dropdown:** A dropdown menu is open, showing the selected field `http_request.response_size` and a "Sort order" dropdown set to `Descending`.

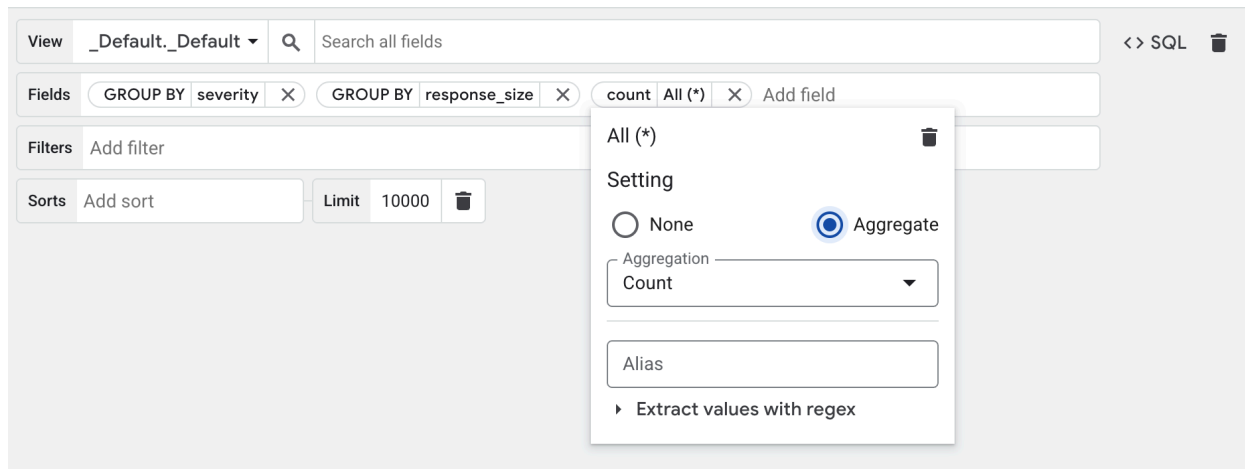
### Corresponding manual query

```
SQL
SELECT
  severity,
  http_request.response_size
FROM
  `query-builder-test-project.global._Default._Default`
GROUP BY
  severity,
  http_request.response_size
ORDER BY
  http_request.response_size DESC
```

### Expected Results

Row	severity	response_size
	STRING	INTEGER
1	NOTICE	3076
2	DEBUG	3075
3	NOTICE	Null

## Example 9: Some number of fields with groupings and a COUNT aggregation



### Corresponding manual query

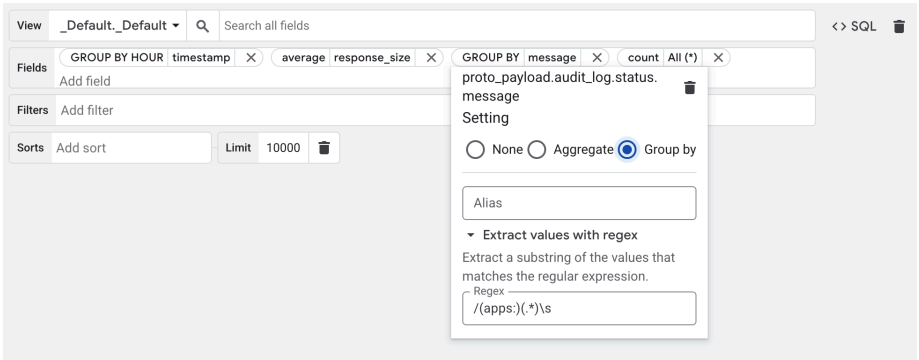
SQL

```
SELECT
  severity,
  proto_payload.request_log.response_size as response_size,
  count(*) as count
FROM
  `query-builder-test-project.global._Default._Default`
GROUP BY
  severity,
  response_size
LIMIT
  1000
```

### Expected results

Row	severity	response_size	count
	STRING	INTEGER	INTEGER
1	NOTICE	3075	274
2	DEBUG	<i>Null</i>	518816
3	WARNING	<i>Null</i>	14720

Example 10: Filter and some number of fields with aggregations including aliases, regexp, COUNT, grouping, sort, and search term



Corresponding manual query

```
SQL
SELECT
  TIMESTAMP_TRUNC(timestamp, HOUR) AS timestamp,
  AVG(http_request.response_size) AS response_size,
  REGEXP_EXTRACT( proto_payload.audit_log.status.message, '/apps:.*\\s') AS
message_failure,
  COUNT(*) AS count
FROM
  query-builder-test-project.global._Default._Default
GROUP BY
  timestamp,
  proto_payload.audit_log.status.message
ORDER BY
  AVG(http_request.response_size) DESC
```

Expected Results

Row	timestamp	response_size	message_failure	count
	TIMESTAMP	FLOAT	STRING	INTEGER
1	2025-05-02 09:00:00.000 EDT	849.8276422764229	app-1-test not found	979292
2	2025-05-02 10:00:00.000 EDT	825.1137295081982	Null	978058
3	2025-05-02 02:00:00.000 EDT	814.1943597560974	Null	456324

## Example 11: Regex extraction - the top 20 files accessed

The following construct shows the top 20 files accessed and shows the use of regex extraction that matches string patterns of “%access\_log”.

The screenshot shows a query builder interface with the following settings:

- View:** \_Default.\_Default
- Fields:** GROUP BY file, count, All (\*)
- Filters:** type = gce\_instance, log\_name LIKE %access\_log
- Sorts:** Add sort
- Limit:** 20

The screenshot shows the 'file' field settings open, with the following configuration:

- Setting:** Group by (selected)
- Alias:** file
- Extract values with regex:** ☒ (checked)
- Regex:** .\* (apache\_access.\*)\$

### Corresponding manual query

```
SQL
SELECT
  REGEXP_EXTRACT(JSON_VALUE(json_payload.file), r".*(apache_access.*)$") as
  file,
  COUNT(*) as count,
FROM
  `query-builder-test-project.global._Default._Default`
WHERE
  resource.type = "gce_instance" AND
  log_name LIKE "%access_log"
GROUP BY file
LIMIT 20
```

### Expected result:

Row	file	count
	STRING	INTEGER

## Supported Comparison Operators

- =
- !=
- = regex
- !=regex
- >
- >=
- <
- <=
- IS NULL
- IS NOT NULL
- LIKE

## Not Supported Features

List of features current version of query builder does not support in the preview:

### Coming Soon

- Cast value type

### To be scheduled

- OR operator in filter
- Partition clause in fields
- Array as input
- TIMESTAMP
  - Filter comparisons
  - FORMAT\_TIMESTAMP
  - Timestamp\_SUB
- Subquery
- IN, NOT IN comparison
- Use regex extraction, aggregation in filter or sort
- Having clause in filter

## FAQ

### # Can I use log scope?

Yes. Log scope is available in query builder. If you need to write more complex query to the same log scope, you can edit your own query by using the log scope CTE generated by query builder.

# Can I save the query builder construct?

No. Builder UI construct cannot be saved.

# Can I join tables?

No. Join is not supported. But you can use the SQL editor to write SQL.

# Does it support "two way mapping" if I write a SQL query?

No. Query builder will auto generate SQL query for valid construct, but it does not support translate your query into builder.