Using The Query Monitoring API

The query monitoring API is a public REST API that allows organization administrators to monitor query activity for a specific AtScale organization. It can be used with any REST-capable monitoring tool to detect anomalies in query volume or response times. This page explains the input parameters, response fields, and recommended client-side monitors. For a short summary of the API, see the Monitoring API Reference.

Host, Ports, And Protocols

Use the fully qualified domain name (FQDN) of the AtScale host or load balancer to access the AtScale Authentication and Engine services. The API will work over the HTTP protocol, however HTTPS is recommended. The default ports for the authetication and engine services are:

Table 1. Service Ports

Service	Port
Authentication	10500
Engine API	10502

Authentication

Token-based Authentication

Because unencrypted credentials are passed over the network, HTTPS / TLS should be used when requesting the bearer token.

1. Request a bearer token. For example, given the user "monitor" and the password "pwd" type the following in a bash shell:

```
export BearerToken=`curl -u monitor:pwd -X GET https://atscale.acme.com:10500/default/auth`
```

2. Save the token for use with subsequent API requests.

Authorization

The following AtScale run-time permissions are required to call this API:

1. Login and View Projects

2. View Queries

Note: It is recommended that you create an AtScale user specifically for the purpose of calling the query monitoring API. Using an admin account to perform monitoring is strongly discouraged.

Request Method

The API is accesssed by making an HTTP GET request to the engine end-point. The {orgId} URI parameter refers to the AtScale Organization Id.

```
GET /monitoring/queries/orgId/{ orgId}
```

Request Headers

The GET request must submit the following request headers.

Table 2. Request Headers

Attribute	Type	Format	Description
Authorization	String	Bearer {BearerToken}	Substitute the bearer token returned from a successful authentication request in the place of {BearerToken}
Content-Type	String	application/json	Specifies the request as a json application request.

Parameters

Table 3 lists the URI query parameters. These parameters are passed according to the URI standard.

Table 3. URI Query Parameters

Parameter Name	Type	Required	Meaning
queryStartTimeFrom	Timestamp	Yes	Return records with query received_time fields greater or equal to this value.
queryStartTimeTo	Timestamp	No	Return records with received_time fields less than this value. If missing, no less-than constraint is applied to the field.
maxInboundQueries	Integer	Yes	The response record limit (in-bound queries). The actual number of returned records is the minimum of this value and the system max limit. See response field result_set_limit for the actual limit used.
includeQueryText	Boolean	No Default = False	If true return the query text in the API response. If False, the query text is omitted.

** Note the queryStartTimeFrom and queryStartTimeTo parameters are in the same time zone as the AtScale server's time zone, which must be set to UTC.

Request Example

The following example illustrates requesting the API with bash and curl. The request is made using the GET method and the bearer token stored in a variable named "BearerToken". The call returns all queries for the "default" organization since 2019-01-31T23:50:59.081Z (UTC time), excluding the query text, up to a maximum of 500 records.

```
curl -s -X GET "https://atscale-engine.acme.com:10502/monitoring/queries/orgId/default?queryStartTimeFrom=2019-01-31T23:50:59.081Z&maxInboundQueries=500&includeQueryText=True" -H "authorization: Bearer ${BearerToken}"
```

Response Body

The API returns detailed information about the processing of each in-bound query as well as each of its outbound queries. Tables 4-7 detail each response field.

Table 4. Global Response Fields. These fields are returned once for each API call.

Attribute	Type	Description
result_set_limit	Integer	Returns the limit applied to the result set. This allows the monitoring client to determine if they are hitting the system limit that may be lower than the maxInboundQueries parameter.

Table 5. In-Bound Query Response Attributes. These fields are returned for each in-bound query.

Attribute	Type	Description
query_id	String	The in-bound Query Id. Cannot be Null. Example: 7a331ad1-b6b5-4812-98a8-130a8c658838
user_id	String	The user login who ran the query. Null if query was run by the system. Example: 'John.Smith'
org_id	String	The id of the org the cube belongs to. Cannot be Null. Example: default
project_id	String	The id of the project the cube belongs to. Cannot be Null. Example: AccountingProject

cube_id	String	The cube id
cube_name	String	The cube name
query_text	String	Inbound query text. Null by default unless input parameter queryBody = True.
failure_message	String	Failure message if the query failed. Null if query succeeded.
received_time	String	Timestamp of the query received event. String type with the following Date time formatting including timezone: 2018-11-14T13:50:01.830Z
started_planning	String	Timestamp of the startPlanning event. String type with the following Date time formatting including timezone: 2018-11-14T13:50:01.830Z
finished_planning	String	Timestamp of the finishedPlanning event. String type with the following Date time formatting including timezone: 2018-11-14T13:50:01.830Z
query_finished	String	Timestamp of the inboundQueryFinished event. String type with the following Date time formatting including timezone: 2018-11-14T13:50:01.830Z
subqueries	List	List of subquery objects. Can be empty if in-bound query failed. See section 'Outbound

		Query Response Attributes' for details.
succeeded	Boolean	True if succeeded, False if failed. Cannot be Null.
aggregate_definition_ids	List	List of strings, For example, ['f43b29c3-299e-49ad-a372-b184834b46bf', 'a952a8bc-299e-49ad-a372-b184834b46bf']
query_language	String	Inbound query type: MDX, SQL, or NONE
used_aggregate_cache	Boolean	True if the query used an in-memory aggregate table, otherwise false.
query_pre_planning_duration	Float	Time spent waiting for query to be prepared for planning. (Includes waiting and parsing). Null if started_planning is Null. (Seconds)
query_planning_duration	Float	Inbound query is planned, resulting in out-bound queries. Null if finished_planning or started_planning have not occurred. (Seconds)
query_wall_duration	Float	The time duration of the completed query. Includes all query processing steps and wait time. Null if the query is still running. (Seconds)
query_wall_running_duration	Float	Includes all steps including wait time and result streaming. If the query is still running, then now() is used instead of query_finished to

		compute the field. For complete queries this field is the same as query_wall_duration. (Seconds)
subqueries_first_wait_duration	Float	Wait time of the first subquery to start execution. This field is Null if no subqueries started execution. (Seconds)
subqueries_wall_duration	Float	Time duration spent running subqueries and returning results. This field is Null if the query is not complete. (Seconds)
subqueries_wall_running_duration	Float	Includes all subquery steps including wait time. If there are running subqueries, now() is used instead of max(subqueryFinished) to compute the field. For complete subqueries this field is the same as subqueries_wall_duration. (Seconds)
subqueries_results_processing_duration	Float	Time spent processing all out-bound queries and streaming results to client. Field is Null if these events have not occurred. (Seconds)

Table 6. Out-Bound Query Response Attributes. These fields are returned for each out-bound query.

Attribute	Type	Description
query_id	String	The out-bound Query Id. Cannot be Null. Example: 7a331ad1-b6b5-4812-98a8-130a8c658838

query_text	String	Out-bound query text. Null by default unless input parameter queryBody = True.
query_failure_message	String	Failure message if the query failed. Null if query succeeded.
succeeded	Boolean	True if succeeded, False if failed. Cannot be Null.
started	String	Timestamp of the subqueryStarted event. String type with the following Date time formatting including timezone: '2018-11-14T13:50:01.830Z'
subquery_fetch_start	String	Timestamp of the subqueryFetchStart event. String Type with the following Date time formatting including timezone: '2018-11-14T13:50:01.830Z'
finished	String	Timestamp of the subqueryFinished event. String type with the following Date time formatting including timezone: '2018-11-14T13:50:01.830Z'
dialect	String	Dialect of the out-bound query, Example, 'Impala-2.7'
used_local_cache	Boolean	True if the query response was served from cache.
is_canary	Boolean	True if the query is a canary query.
		Extra properties of the

extra_props	String	subquery. Currently only Returns the GBQ query id. Only applicable when connected to Google BigQuery. Can be Null.
subquery_wait_duration	Float	Time spent waiting for a prerequisite query or connection to data warehouse. Includes time waiting for other required subqueries, connection pool connection, connection set-up statements, connection liveness test. (Seconds)
subquery_exec_duration	Float	Amount of time query takes to execute. This field is Null if the query is not complete. (Seconds)
subquery_fetch_results_duration	Float	Time to fetch subquery results. This field is Null if subquery fetching is not complete. (Seconds)
subquery_wall_duration	Float	The time duration of the completed subquery. Includes all query processing steps and wait time. This field is Null if the query is not complete. (Seconds)
subquery_wall_running_duration	Float	Includes all steps including wait time. If the subquery is not complete, then uses now() instead of the finished field in the calculation. For complete queries this field is the same as subquery_wall_time_duration. (Seconds)

Client-Side Query Monitoring Statistics

The API provides query performance statistics and event timestamps for each in-bound and out-bound query. Because this is a very fine level of detail you will want to aggregate the data on the client-side for monitoring purposes. For

example, you are likely interested in monitoring the count of queries or the mean query time over the last 5 minutes. The following section contains suggested client-side calculations, alerts, and links to detailed Nagios examples.

Table 7. Suggested Client-Side Monitoring Statistics ("s.t." = "such that")

Metric Name	Calculation	Description
inboundQueryCount	count(in-bound_queryId)	Total number of in-bound queries
successfulQueryCount	count(in-bound_queryId), s.t. succeeded=True	Total number of successful Queries
queryErrorRate	if(inboundQueryCount == 0, 0, (inboundQueryCount - successfulQueryCount) / inboundQueryCount * 100)	Percentage of in-bound queries that failed
meanQueryTime	sum(query_wall_duration) / inboundQueryCount	Mean total query time (take care to exclude Null values for running queries)
maxTotalQueryTime	max(query_wall_running_duration)	Max total query time (includes running queries)
queryID_maxTotalQueryTime	queryID s.t. query_wall_running_duration = maxTotalQueryTime	Query ID of the query with the maximum total Query time
maxFirstSubQueryWait	max(subqueries_first_wait_duration)	Maximum first subquery wait time. Take care to omit Null values.
meanFirstSubQueryWait	avg(subqueries_first_wait_duration)	Mean first subquery wait time. Take care to omit Null values.
maxInboundQueryPlanning	max(query_planning_duration)	Maximum in-bound query planning time.

queryID_maxInboundQueryPlanning	queryID s.t. query_planning_duration = max(query_planning_duration)	Query ID of the query with the maximum inboundQueryPlanning value
out-boundQueryCount	count(sqei.queryId)	Total number of out-bound queries
out-boundCachedQueryCount	count(sqei.queryId) s.t. used_local_cache = true	Total number of out-bound queries served from cache

There are a large number of statistics that could be monitored, however Table 8 lists the basic alert conditions that should be monitored and the suggested corresponding task for the Organization's Administrator. The alarm thresholds are suggested values and must be tuned for the environment being monitored. Finally the remediation actions are general suggestions for typical bottlenecks. The actual inveatigation task will depend on the specific data warehouse you are using and your reporting use-case.

Nagios plugin examples for the alerts listed in Table 8 are available on the AtScaleInc/apidemo GitHub repo. You may request access to this repo by filing a support request.

Table 8. Example Alert Conditions and Response Actions

Example Alert	Org Admin Alert Response Actions
maxTotalQueryTime > 120 s	If this alarm happens in conjunction with other system alarms then start by checking shared resources, such as the data warehouse's processing queue. If the alarm is restricted to a small number of queries, then search for the queries in Design Center using queryID_maxTotalQueryTime. Ask the query's users if its complexity is warranted, run explain plan in Design Center, check aggregate usage, partition pruning, and ask the data warehouse admin about the cluster stats on the tables used by the query(ies).
queryErrorRate > 1%	Go to Design Center and search for failed queries at the time of the alarm for more information. If the error is a connection timeout error then contact the data warehouse administrator or network administrator.
maxFirstSubQueryWait > 5 s	Check if any of the following are true: 1 .Data warehouse admission control settings are too low, 2. Data warehouse is overloaded, 3. inboundQueryCount has spiked, 4. out-boundQueryCount has spiked. 5. Hive metstore is slow. 6. AtScale connection pool is too small for concurrent inboundQueryCount. 7. data warehouse loadbalancer is using sticky sessions, 8. data warehouse is under-powered for the load. 9. If Kerberos and DA is enabled, the DB Connection pool may be cold and take a while to warm-up.
maxInboundQueryPlanning > 3 s	Go to Design Center and search for the query using queryID_maxInboundQueryPlanning. Contact the cube owner to confirm the need for the level of complexity (Number of fields in select, size of cube, number of aggregates (possibly too high), perform a workbook best practices review)

Example Nagios Plugin

Any REST-enabled monitoring tool can access the AtScale query monitoring API. Example 1 illustrates how a bash script can be used as a Nagios plugin to detect the Table 8. alarm condition of "maxFirstSubQueryWait > $threshold".

This example uses the following Unix command line utilities:

- base64
- curl
- jq

In this example the script parameters are:

- $1 = Fully Qualified Domain Name (FQDN) of the AtScale Host or load balancer (string).
- $2 = Threshold value in seconds for the maximum duration for maxFirstSubQueryWait value (seconds, integer).
- $3 = Time offset to use for the queryStartTimeFrom constraint (minutes, integer). This should equal the script polling interval.
- $4 = max query results to request
- $5 = user id for the monitoring user
- $6 = user id's password

The details of each step are provided as comments in-line.

Example 1. Bash Script Nagios plugin that Checks the "maxFirstSubQueryWait" Condition from Table 8.

```
#!/bin/bash
# Nagios Plugin Bash Script - check_first_subquery_wait_time.sh
# This script checks if the AtScale first subquery wait time exceeds a threshold.

if [[ "$#" -ne 6 ]]; then
   echo "Incorrect number of parameters! Syntax: ./check_first_subquery_wait_time.sh host threshold_percent
start_minutes_past max_queries user password"
   exit 3
fi

host=$1
threshold=$2
startTimeMinutesPast=$3
maxQueryCount=$4
user=$5
password=$6

startTimeCmd="date --utc +%FT%T.00Z --date='$startTimeMinutesPast minutes ago'"
startTime=`eval $startTimeCmd`

atscaleCmd="'http://$host:10502/monitoring/queries/orgId/default?
queryStartTimeFrom=$startTime&maxInboundQueries=$maxQueryCount'"

jqCmd="jq '[.response.data[].subqueries_first_wait_duration] | max'"

jwtCmd="curl -X GET -u $user:$password 'http://$host:10500/default/auth'"

jwt=`eval $jwtCmd`

curlCmd="curl -v -X GET -H 'Content-Type: application/json' -H 'Authorization:Bearer $jwt' $atscaleCmd | $jqCmd"

result=`eval $curlCmd`

if (( $(echo "$result < $threshold" |bc -l) )); then
   echo "OK, First Subquery Wait Time ($result) is less than $threshold"
   exit 0
else
   echo "CRITICAL, First Subquery Wait Time ($result) is greater than or equal to $threshold"
   exit 2
fi
```

Nagios plugin examples for the alerts listed in Table 8 are available on the AtScaleInc/apidemo GitHub repo. You may request access to this repo by filing a support request.