Administration & Operations

AQL

AEROSPIKE

aql – Aerospike Query Language

Server Access

```
$ ssh aerotraining@ip_addr
password: aerotraining

$sudo service aerospike start
$sudo service amc start

(stop, status, restart are other options)
```

aql – Command Line Interface

AQL is the Aerospike command line tool for managing the database with SQL–**like** commands. Major functions include:

- Record Operations
- Querying Records
- Data Management
- Security Management (User/Roles – Enterprise only)
- Index Management
- Query Scan Management
- Settings / Statistics
- UDF Management

aql - Syntax

Common Usage:

```
aql OPTIONS
```

Option	Default	Description
-h	127.0.0.1	Seed host to connect to. AQL will learn about the other nodes in the cluster from this one.
-p	3000	Seed port.
-c	[none]	Command to run.
-f	[none]	Execute the commands in the specified file.
-o	table	One of “json” or “table”. The output format of queries.
--help	[none]	Display usage information.

In most cases, you will enter the `aql` CLI and enter additional commands (unless running a script with the –f command). You will then see the aql prompt “aql> “.

aql - Help

Use HELP command to see summary of available AQL commands and their sample usage.

```
aql> HELP
```

Get familiar with commands to:

- Perform CRUD Ops on Records
- Manage Secondary Indexes
- SHOW info on namespaces, sets, bins
- Manage UDFs
- Set / Get Parameters local to current AQL session

aql - Write Test

Let us now conduct a simple write test on the database. The default configuration of Aerospike has a simple RAM based namespace called "test". Let's try to write a record into a set called "testset".

From the command line of the server, run the aql command.

```
$ aql
```

```
$ aql> INSERT INTO test.testset (PK, company, age) \
       VALUES('myname', 'mycompany', '35')
```

Literals in AQL such as ‘test’ are case sensitive.

Command words like ‘INSERT’ are not case sensitive.

aql - Read / Delete Tests

To read the data from the command line, you must be in the aql shell.

```
aql> SELECT * from test.testset where PK='myname'
```

This should give you back the values you wrote in the previous step.

For more information on a record, use the “explain” feature (3.6.1+).

```
aql> EXPLAIN SELECT * from test.testset where PK='myname'
```

Finally, you should be able to delete the record. From the aql command line, run the following:

```
aql> DELETE FROM test.testset WHERE PK='myname'
```

aql – Record Access Using Record Digest

You can read a record given its DIGEST in Hexadecimal or EDIGEST in Base64. (These are not yet documented in AQL Help.)

EXPLAIN command can be used to see the DIGEST in Hexadecimal.

Setting output to JSON shows digest in Base64. `(aql>set output json)`

To read the data from the command line, you must be in the aql shell.

```
aql> SELECT * from test.testset where DIGEST='AE0134BB.....'
OR
aql> SELECT * from test.testset where
EDIGEST="Ae0134bB.....0k="
```

This should give you back the values you wrote in the previous step.

aql – Manage Secondary Indexes

CREATE INDEX <index> ON <ns>[.<set>] (<bin>) NUMERIC|STRING|GEO2DSPHERE

CREATE LIST/MAPKEYS/MAPVALUES INDEX <index> ON <ns>[.<set>] (<bin>) NUMERIC|STRING|GEO2DSPHERE

DROP INDEX <ns>[.<set>] <index>
REPAIR INDEX <index> ON <ns>[.<set>]

<ns> is the namespace for the index.
<set> is the set name for the index.
<index> is the name of the index.

Examples:

CREATE INDEX idx_foo ON test.demo (foo) NUMERIC
DROP INDEX test.demo idx_foo
REPAIR INDEX idx_foo ON test.demo

aql – Manage User Defined Functions (UDFs)

MANAGE UDFS
REGISTER MODULE '<filepath>'
SHOW MODULES
REMOVE MODULE <filename>
DESC MODULE <filename>

<filepath> is file path to the UDF module(in single quotes).
<filename> is file name of the UDF module.

Examples:

REGISTER MODULE '~/test.lua'
SHOW MODULES
DESC MODULE test.lua
REMOVE MODULE test.lua

aql – Invoke User Defined Functions (UDFs)

EXECUTE <module>.<function>(<args>) ON <ns>[.<set>]
EXECUTE <module>.<function>(<args>) ON <ns>[.<set>] WHERE PK = <key>
AGGREGATE <module>.<function>(<args>) ON <ns>[.<set>] WHERE <bin> = <value>
AGGREGATE <module>.<function>(<args>) ON <ns>[.<set>] WHERE <bin> BETWEEN <lower> AND <upper>

<module> is UDF module containing the function to invoke.
<function> is UDF to invoke.
<args> is a comma-separated list of argument values for the UDF.
<ns> is the namespace for the records to be queried.
<set> is the set name for the record to be queried.
<key> is the record's primary key.
<bin> is the name of a bin.
<value> is the value of a bin.
<lower> is the lower bound for a numeric range query.
<upper> is the lower bound for a numeric range query.

Examples:

EXECUTE myudfs.udf1(2) ON test.demo
EXECUTE myudfs.udf1(2) ON test.demo WHERE PK = 'key1'
AGGREGATE myudfs.udf2(2) ON test.demo WHERE foo = 123
AGGREGATE myudfs.udf2(2) ON test.demo WHERE foo BETWEEN 0 AND 999

aql - basic examples

List all namespaces in the cluster

```
aql> show namespaces
+----------------+
| namespaces     |
+----------------+
| “test”         |
| “bar”          |
+----------------+
```

List all sets in the cluster

```
aql> show sets
+-----------+----------------+----------------------+---------+-----------+------------+----------------------+
| n_objects | set-enable-xdr | set-stop-write-count | ns_name | set_name  | set-delete | set-evict-hwm-count |
+-----------+----------------+----------------------+---------+-----------+------------+----------------------+
| 100000    | "use-default"  | 0                    | "test"  | ”testset" | "false"    | 0                    |
+-----------+----------------+----------------------+---------+-----------+------------+----------------------+
1 row in set (0.001 secs)
OK
```

List all bins in the namespace

```
aql> show bins
+-------+-----+-------+-----------+
| quota | bin | count | namespace |
+-------+-----+-------+-----------+
| 32768 | "0" | 1     | "test"    |
+-------+-----+-------+-----------+
1 row in set (0.000 secs)
OK
```

aql – Useful Settings, RUN command

SETTINGS (Partial List)

TIMEOUT (time in ms, default: 1000) – use to limit output
RECORD_TTL (time in sec, default: 0)
RECORD_PRINT_METADATA (true | false, default false)
OUTPUT (TABLE | JSON, default TABLE)
LUA_USERPATH <path>, default : /opt/aerospike/usr/udf/lua
LUA_SYSPATH <path>, default : /opt/aerospike/sys/udf/lua
KEY_SEND (true | false, default false) - If set true, key added as a bin.
DURABLE_DELETE (true | false, default false)

To get the value of a setting, run:

aql> GET <setting>

To set the value of a setting, run:

aql> SET <setting> <value>

- **aql>RUN "filepath/filename"** -- handy to run a set of AQL commands in a text file.