Contents

MongoDB Schema Manager

Overview

MongoDB Schema Manager is a tool for generating schemata for MongoDB collections. These schemata are crucial for proper operation of the MongoDB SQL interface because it is strongly typed and must know the types of fields in your collections for all but the most simple of queries.

The schemata generated by the MongoDB Schema Manager are the same as those supported by MongoDB collections

Installation

Copy the `mongodb-schema-manager` executable to your desired location. On some systems, it may be necessary to give the binary executable permissions

```
chmod +x mongodb-schema-manager
```

Usage

The general usage of mongodb-schema-manager (given the name for your platform):

mongodb-schema-manager [OPTIONS]

The options are:

Options:

- -f, --file <CONFIG_FILE> The path to the configuration file (optional). If not provided, you must specify --uri to connect to a MongoDB cluster
  - the config file must be specified in yaml and specify some or all of the following options, with at least uri present. An example of a valid config file can be seen can be seen in the next section
- --uri The cluster URI (optional). If not provided, you must specify --file to load a configuration file with the URI
- -u, --username Username for authentication (optional). Not all authentication mechanisms require username.
- -p, --password Password for authentication (optional). Not all authentication mechanisms require password, and it is generally safer to specify in the config file so that it does not appear on task lists
- --ns-include <NS_INCLUDE> The databases and collections to include (optional). If not provided, all databases and collections are included
- --ns-exclude <NS_EXCLUDE> The databases and collections to exclude (optional). If not provided, no databases or collections are excluded
- --quiet Enables quiet mode for less output
- -o, --logpath Log directory path where to write log files. If not provided, logging will be ignored
- -v, --verbosity The logging level to capture in the log file (optional). Requires logpath to be set [default: warn] [possible values: trace, debug, info, warn, error]\
- -a, --action <SCHEMA_ACTION> Specifies the action to execute on the schema [default: merge] [possible values: overwrite, merge, clear]
- --dry-run Perform a dry run without analyzing schema or writing to the database. Useful for testing ns-include and ns-exclude
- --resolver Resolver option for DNS resolution (optional). Specify a resolver if DNS resolution fails or takes too long [possible values: cloudflare, google, quad9]
- -h, --help Print help (see more with '--help')
- -V, --version Print version

Valid Configuration File

```yaml
uri: mongodb://test:test@localhost:27017/
ns-include:
  # Include all collections in only the database config_db.
  - config_db.*
ns-exclude:
  # But do not include the foo collection. Note, namespaces do not need to be specified
  # in ns-include prior to specifying them in ns-exclude.
  - config_db.foo
quiet: false
dry-run: false
```

Authentication Mechanisms

Authentication Mechanisms currently supported by the mongodb-schema-manager are most of those supported by the MongoDB Rust driver on which the mongodb-schema-manager is built. The only exception at this point is the Workforce flow for OIDC. A more in-depth representation of supported Authentication mechanisms can be seen here

Briefly, MongoDB Authentication Mechanisms have 5 attributes that may, must, or must not be set based on the chosen Mechanism. These are:

- mechanism: the name of the Authentication Mechanism itself.
- username: the username for the user connecting to the database, perhaps surprisingly, some forms of authentication actually do not use this.
- password: this is password for the user.
- source: this is the database against which Authentication should be performed. In many mechanisms, this will be `$external` which means that Authentication is actually performed outside of MongoDB. Many other mechanisms allow specific credential information to be set in specific databases, though in practice this is often just the `admin` database.
- mechanism_properties: these are additional key values pairs used by some select Mechanisms.

These attributes are set in the uri passed for connection, a guide on how to set these attributes in the uri can be seen here, the entire documentation for uris can be found here

Mechanisms:

- SCRAM-SHA-1: Allows for connecting with username and password protected using SCRAM-SHA-1
  - mechanism MUST be "SCRAM-SHA-1"
  - username MUST be specified and non-zero length.
  - password MUST be specified.
  - source MUST be specified. Defaults to the database name if supplied on the connection string or admin.
  - mechanism_properties MUST NOT be specified.
- SCRAM-SHA-256: Allows for connecting with username and password protected using SCRAM-SHA-256
  - mechanism MUST be "SCRAM-SHA-256"
  - username MUST be specified and non-zero length.
  - password MUST be specified.
  - source MUST be specified. Defaults to the database name if supplied on the connection string or admin.
  - mechanism_properties MUST NOT be specified.
- MONGODB-X509: Allows for connecting via an x509 certificate, see the MongoDB documentation for more information.
  - mechanism MUST be "MONGODB-X509"
  - username SHOULD NOT be provided for MongoDB 3.4+ MUST be specified and non-zero length for MongoDB prior to 3.4
  - password MUST NOT be specified.
  - source MUST be `$external`. Defaults to `$external`.
  - mechanism_properties MUST NOT be specified.
- GSSAPI (kerberos)

  - mechanism MUST be "GSSAPI"
  - username MUST be specified and non-zero length.
  - source MUST be "$external". Defaults to $external.
  - password MAY be specified. If omitted, drivers MUST NOT pass the username without password to SSPI on Windows and instead use the default credentials.
  - mechanism_properties: -- SERVICE_NAME may allow the user to specify a different service name. The default is "mongodb". -- CANONICALIZE_HOST_NAME may allow the user to request canonicalization of the hostname. This might be required when the hosts report different hostnames than what is used in the kerberos database. The value is a string of either "none", "forward", or "forwardAndReverse". "none" is the default and performs no canonicalization. "forward" performs a forward DNS lookup to canonicalize the hostname. "forwardAndReverse" performs a forward DNS lookup and then a reverse lookup on that value to canonicalize the hostname. The service will fallback to the provided host if any lookup errors or returns no results. -- SERVICE_REALM may allow the user to specify a different realm for the service. This might be necessary to support cross-realm authentication where the user exists in one realm and the service in another. -- SERVICE_HOST may allow the user to specify a different host for the service. This is stored in the service principal name instead of the standard host name. This is generally used for cases where the initial role is being created from localhost but the actual service host would differ.

- MONGODB-AWS
  - mechanism MUST be "MONGODB-AWS"
  - username MAY be specified. The non-sensitive AWS access key.
  - password MAY be specified. The sensitive AWS secret key.
  - source MUST be `$external`. Defaults to `$external`.
  - mechanism_properties
    - AWS_SESSION_TOKEN allows the user to specify an AWS session token for authentication with temporary credentials.

- MONGODB-OIDC: Allows for connecting via OpenID Connect access tokens. Currently, only Workload authentication in the azure or gcp environments and Workforce authentication are supported
  - username MAY be specified. Its meaning varies depending on the OIDC provider integration used.
  - source MUST be `$external`. Defaults to `$external`.
  - password MUST NOT be specified.
  - mechanism MUST be "MONGODB-OIDC"
  - mechanism_properties
    - ENVIRONMENT allows the user to specify the name of a built-in OIDC application environment integration to use to obtain Workload credentials. The value MUST be one of ["azure", "gcp"].
      - If not set, Workforce authentication is assumed.
    - TOKEN_RESOURCE (currently required, if ENVIRONMENT is set) the URI of the target resource.

- PLAIN (LDAP)
  - mechanism MUST be "PLAIN"
  - username MUST be specified and non-zero length.
  - password MUST be specified.
  - source MUST be specified. Defaults to the database name if supplied on the connection string or `$external`.
  - mechanism_properties MUST NOT be specified.

Examples

The easiest mode in which to use the mongodb-schema-manager is connecting to a local cluster with username and password:

```
mongodb-schema-manager --uri mongodb://username:password@localhost
```

The mongodb-schema-manager will output the schemata namespaces generated:

*Schema creation has completed successfully. Now printing all namespaces with created or modified schemas: Database: foo2. Namespaces: ["coll"]. Database: oncall. Namespaces: ["personneljob"]. *Database: zugzugs. Namespaces: ["zugzug_result_13552373-6308-4972-a277-d6fa0ed8f81e", "zugzug_result_4bee6bff-5ccb-41ad-96d4-*

c96ea43205de", "zugzug_result_92d8e06a-c538-4b98-b83b-a25c24bce201", "zugzug_result_cc8ffd89-afda-49a9-9fd1-705e6574f367", "zugzugs"].

Alternatively, one can specify a config file:

```
mongodb-schema-manager -f ./config.yml
```

Using Azure IMDS oidc for workload can be achieved as follows:

mongodb-schema-manager --uri mongodb://somecluster.a.query.mongodb.net/?ssl=true&authMechanism=MONGODB-OIDC&authMechanismProperties=ENVIRONMENT:azure,TOKEN_RESOURCE:http://example.com

The ns-include and ns-exclude options can be used to include and exclude namespaces as desired:

mongodb-schema-manager --uri mongodb://localhost --ns-include zugzugs. --ns-exclude *.zugzug_result_92d8e06a-c538-4b98-b83b-a25c24bce201*

Results in:

Schema creation has completed successfully. Now printing all namespaces with created or modified schemas: Database: zugzugs. Namespaces: ["zugzug_result_13552373-6308-4972-a277-d6fa0ed8f81e", "zugzug_result_4bee6bff-5ccb-41ad-96d4-c96ea43205de", "zugzug_result_cc8ffd89-afda-49a9-9fd1-705e6574f367", "zugzugs"].

Schema Instability

The mongodb-schema-manager looks for what we call schema instability. Schema instability occurs when too many fields appear that differ from document to document in the collection. This is a general design pattern some MongoDB users use where the documents are treated as hash maps instead of as structured data. When this occurs, the mongodb-schema-manager will cap the number of fields in the collection schema and produce a warning. This design pattern is not acceptable for SQL because each top level field is treated as a column in SQL.