Introduction

Eman Fathi

Information Technology Institute

Document Validations version 3.2

jsonSchema Validation version3.6

Document Validation

- ✓ Introduced in MongoDB3.2
- ✓ MongoDB provides the capability to validate documents during updates and insertions. Validation rules are specified on a per-collection basis using the validator option, which takes a document that specifies the validation rules or expressions.

To specify document validation on a new collection, use the new validator option in the db.createCollection() method.

To add document validation to an existing collection, use the new validator option in db.runComm() method

Validation occurs during updates and inserts. Existing documents do not undergo validation checks until modification.

To view the validation specifications for a collection, use the db.getCollectionInfos() method.

Json Schema Validation

- Introduced in MongoDB 3.6
- a “Validation Schema” was already introduced in 3.2 but the new “JSON Schema Validator” introduced in the 3.6 release is by far the best and a friendly way to manage validations in MongoDB.
- JSON Schema Validation extends Document Validation in many different ways, including the ability to enforce schemas inside arrays and prevent unapproved attributes from being added.
- To specify document validation on a new collection, use the new validator option in the db.createCollection() method.
- To add document validation to an existing collection, use the new validator option in db.runCommand() method
- To view the validation specifications for a collection, use the db.getCollectionInfos() method.

Json Schema Validation

To specify certain fields to be exists and can have extra filed

```
db.runCommand( {
   collMod: "customers",
   validator: { $jsonSchema: {
      bsonType: "object",
      required: [ "phone", "name" ],
      properties: {
         phone: {
            bsonType: "string",
            description: "must be a string and is required"               },
         name: {
            bsonType: "string",
            description: "must be a string and is required"
         }
      }
   } },
   validationLevel: "moderate"
} );
```

Json Schema Validation

To specify all files needed as optional and can have extra filed

```
db.runCommand( {
   collMod: "customers",
   validator: { $jsonSchema: {
      bsonType: "object",
      properties: {
         phone: {
            bsonType: "string",
            description: "must be a string and is required"
               },
         name: {
            bsonType: "string",
            description: "must be a string and is required"
         }
      }
   } },
   validationLevel: "moderate"
} );
```

Json Schema Validation

To specify certain fields (additionalProperties) without adding additional ones
But _id must be specified

```
db.runCommand( {
   collMod: "customers",
   validator: { $jsonSchema: {
      bsonType: "object",
      additionalProperties:false,
      required: [ "_id","phone", "name" ],

      properties: {
         _id:{},
         phone: {
            bsonType: "string",
            description: "must be a string and is required"
         },
         name: {
            bsonType: "string",
            description: "must be a string and is required"
         }
      }
   } }
} );
```

Json Schema Validation

Specifying more data types

```
price: {
  bsonType: "decimal",
  description:"'price' must be a decimal and is required"
},
quantity: {
  bsonType: ["int", "long"],
  minimum: 1,
  maximum: 100,
  exclusiveMaximum: true,
  description:"'quantity' must be short or long integer between 1 and 99"
},
major: {
  enum: [ "Math", "English", "Computer Science", "History", null ],
  description: "can only be one of the enum values and is required"
    }
```

https://docs.mongodb.com/manual/reference/bson-types/

Document Validation

validationAction :Determines whether to error on invalid documents or just warn about the violations but allow invalid documents to be inserted

.

```
db.customers.runCommand("collMod",{validationLevel: "moderate", validationAction: "warn"});
```

validationAction	Description
"error"	Default Documents must pass validation before the write occurs. Otherwise, the write operation fails.
"warn"	Documents do not have to pass validation. If the document fails validation, the write operation logs the validation failure.

Document Validation

validationLevel :Determines how strictly MongoDB applies the validation rules to existing documents during an update.

```
db.customers.runCommand("collMod",{validationLevel: "moderate", validationAction: "warn"});
```

validationLevel	Description
"off"	No validation for inserts or updates.
"strict"	Default Apply validation rules to all inserts and all updates.
"moderate"	Apply validation rules to inserts and to updates on existing valid documents. Do not apply rules to updates on existing invalid documents.