**MongoDB Notes**

MongoDB is a No-SQL (Not Only SQL) database that is document-oriented without a pre-defined schema.

Main structure of MongoDB:

1. Database: Organized group of collections
2. Collection: Organized group of documents with common fields (like tables)
3. Document: Method to organize and store data as a set of field-value pairs (like rows)



Unlike SQL, MongoDB is built for scaling-out by adding on additional nodes onto a cluster for adding storage and increasing read-write throughput.

Advantages and disadvantages of using MongoDB:

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| Saves time from pre-defining schema | High memory required |
| Easy to scale | Document size has limit of 16mb. |
| Easy to store arrays and objects | Does not support transactions |
| Higher performance than SQL due to use of internal memory for storage |  |

There are 3 ways for connecting with MongoDB:

1. MongoDB Shell (Local Command Line Interface – Main focus)
2. MongoDB Compass (GUI on local drive)
3. MongoDB Atlas (GUI on cloud)

**Documents**

Documents in MongoDB have the following structure:

**{field1: value1, field2: value2, …}**

Caveats of MongoDB documents:

1. MongoDB is type-sensitive and case-sensitive
2. Duplicate fields are not allowed
3. Avoid using reserved characters like “.” and “$” for naming fields

Note that every document has special field “\_id” that is unique within a collection and it is generated by default.

**Collections**

While documents can be placed into any collection, it is best practice to store similar document schema under the same collection for the following reasons:

1. Ensure query of documents only returns results that follow a particular schema
2. Grouping similar documents allow for data locality
3. Better efficiency of indexing collections

When naming collections, the following restrictions apply:

1. Empty string is not valid
2. Collection names should not be reserved keywords like “system”
3. Collection names should not contain reserved characters like “$” and “.”

Note that multiple collections can be created in a single database by using name-spaced subcollections separated by “.” sign.

Example: mydb.coll1 and mydb.coll2 indicates that there are two collections named coll1 and coll2 under a database named as “mydb”.

**Databases**

Databases in MongoDB can also consist of multiple collections.

When naming databases, the following restrictions apply:

1. Empty string is not valid
2. Database names should not have any special characters (Only alphanumeric allowed)
3. Database names are case-insensitive

MongoDB has three reserved database names as the following:

1. **Admin**: Used for database administration process
2. **Local**: Stores data used during replication process
3. **Config**: Stores information about each shard within a MongoDB cluster

**Data Types of MongoDB**

For MongoDB, data types do not need to be defined as MongoDB will automatically identify the data types of inputs.

The following are the lists of data types available for MongoDB:

|  |  |
| --- | --- |
| String | Null |
| Integer | Date (Represent milliseconds since Unix Epoch) |
| Boolean | Object ID (Used in “\_id” field) |
| Double | Binary |
| Arrays (Data contained in arrays do not have to be same data type – Behaves like lists in Python) | Code (Stores Java-Script code) |
| Timestamp | Regular Expression |
| Object (Used for embedded documents and creating dates in proper format) |  |

While JSON only supports a few data types, Binary JSON (BSON) extends the list of supported data types as shown above for efficient encoding and decoding within different languages.

**Basic Commands for MongoDB Shell**

1. Running MongoDB on CLI (On directory with mongo.exe)

**mongo**

1. Viewing current database used

**db**

1. Switch to using another database

**use new\_db**

1. Showing list of all databases

**show dbs**

1. Showing list of all collections for a specific database

**show collections**