MONGODB - GRIDFS

GridFS is the MongoDB specification for storing and retrieving large files such as images, audio files, video files, etc. It is kind of a file system to store files but its data is stored within MongoDB collections. GridFS has the capability to store files even greater than its document size limit of 16MB.

GridFS divides a file into chunks and stores each chunk of data in a separate document, each of maximum size 255k.

GridFS by default uses two collections **fs.files** and **fs.chunks** to store the file's metadata and the chunks. Each chunk is identified by its unique _id ObjectId field. The fs.files severs as a parent document. The **files_id** field in the fs.chunks document links the chunk to its parent.

Following is a sample document of fs.files collection –

```
{
   "filename": "test.txt",
   "chunkSize": NumberInt(261120),
   "uploadDate": ISODate("2014-04-13T11:32:33.557Z"),
   "md5": "7b762939321e146569b07f72c62cca4f",
   "length": NumberInt(646)
}
```

The document specifies the file name, chunk size, uploaded date, and length.

Following is a sample document of fs.chunks document –

```
{
   "files_id": ObjectId("534a75d19f54bfec8a2fe44b"),
   "n": NumberInt(0),
   "data": "Mongo Binary Data"
}
```

Adding Files to GridFS

Now, we will store an mp3 file using GridFS using the **put** command. For this, we will use the **mongofiles.exe** utility present in the bin folder of the MongoDB installation folder.

Open your command prompt, navigate to the mongofiles.exe in the bin folder of MongoDB installation folder and type the following code –

```
>mongofiles.exe -d gridfs put song.mp3
```

Here, **gridfs** is the name of the database in which the file will be stored. If the database is not present, MongoDB will automatically create a new document on the fly. Song.mp3 is the name of the file uploaded. To see the file's document in database, you can use find query –

```
>db.fs.files.find()
```

The above command returned the following document –

```
{
   _id: ObjectId('534a811bf8b4aa4d33fdf94d'),
   filename: "song.mp3",
   chunkSize: 261120,
   uploadDate: new Date(1397391643474), md5: "e4f53379c909f7bed2e9d631e15c1c41",
   length: 10401959
}
```

We can also see all the chunks present in fs.chunks collection related to the stored file with the following code, using the document id returned in the previous query –

```
>db.fs.chunks.find({files_id:ObjectId('534a811bf8b4aa4d33fdf94d')})
```

In my case, the query returned 40 documents meaning that the whole mp3 document was divided in 40 chunks of data.