Database Design – Answers

A possible solution would be to create 2 separate JSON files:

Document type 1:

```
{
  "ArticleID" : "1",
  "UploadedBy" : {
    "UserID" : "1",
    "Name" : "Peter",
    "City" : "Bunnik"
  },
  "Category" : {
    "CategoryID" : "1",
    "CategoryName" : "Sports"
  },
  "UploadDate" : "10-14-2021",
  "URL" : "https://..."
}
```

Document type 2:

```
{
  "CommentID" : "1",
  "UserID" : "2",
  "ArticleID" : "1",
  "Date" : "10-15-2021",
  "Comment" : "Nice ..."
}
```

Both document types can use the ArticleID to partition the data. ArticleID would distribute the data evenly over the cluster. Using the same partition key would make it cheaper to retrieve an article and it's comments because they come from the same partition. With extremely large amounts of comments on a single article that could become a bottleneck. Also when specific articles have a lot of comments and others don't, using the ArticleID might result to skew in your data storage.

When just a couple of comments per article are expected, comments can become a nested array of comment objects within document type 1. But with a lot of comments per article this may result in inefficiently large documents. With comments added over a longer period of time this would result in lots of updates of existing documents which is more expensive than inserting new documents.

Depending on usage and how often user data changes, document 2 can store all user info as a nested object just like document 1 does.