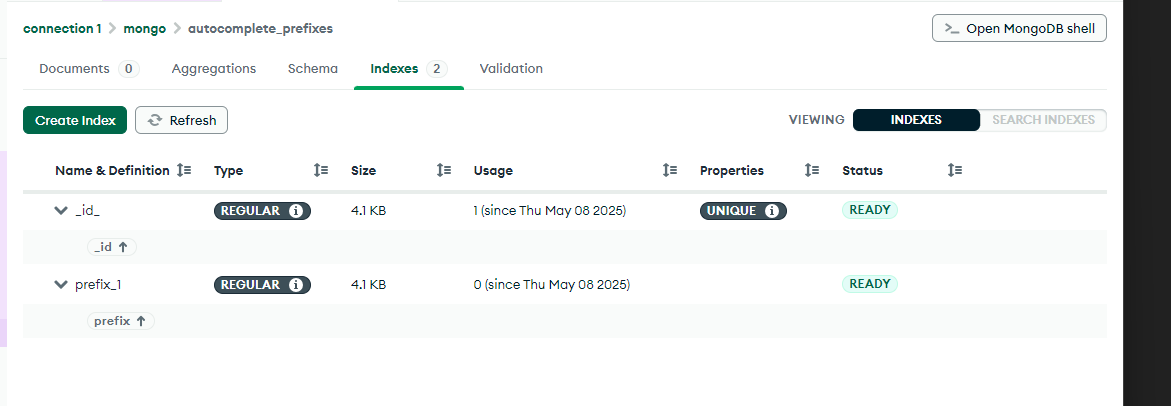
**Step 1: MongoDB Schema & Index (Backend Store)**

Let’s define the **correct MongoDB collection schema** for trending autocomplete:

|  |
| --- |
| {  "prefix": "how to",  "completions": [  {  "query": "how to install linux ",  "frequency": 56,  "last\_updated": "2025-04-29T10:00:00Z"  },  {  "query": "how to use mongodb",  "frequency": 34,  "last\_updated": "2025-04-29T10:00:00Z"  }  ]  } |

|  |
| --- |
| docker exec -it mongodb mongosh  use mymongo  // here we create index on prefix field for fast lookup  db.autocomplete\_prefixes.createIndex({ prefix: 1 }) |



### Algorithm Plan

1. Reads user search queries from an HDFS text file (one query per line).
2. Generates all prefixes of length 2 to 60 from each query.
3. Counts (prefix, query) frequencies.
4. Picks top K completions per prefix.
5. Stores into MongoDB in documents of the form:

|  |
| --- |
| {  "prefix": "...",  "completions": [  { "query": "...", "frequency": 123, "last\_updated": ISODate(...) },  …  ]  } |

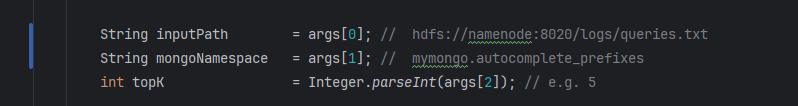
Step -1

Define the argument:

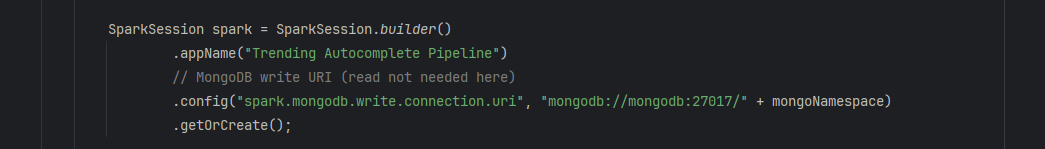
args[0] = HDFS path to your hourly-collected log file

args[1] = MongoDB namespace to write into

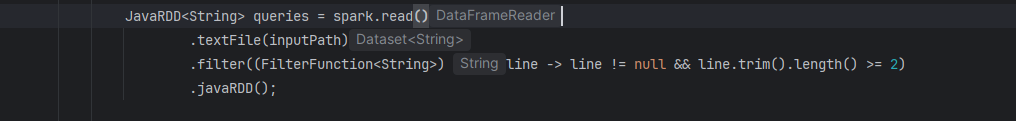
args[2] = top-K completions per prefix



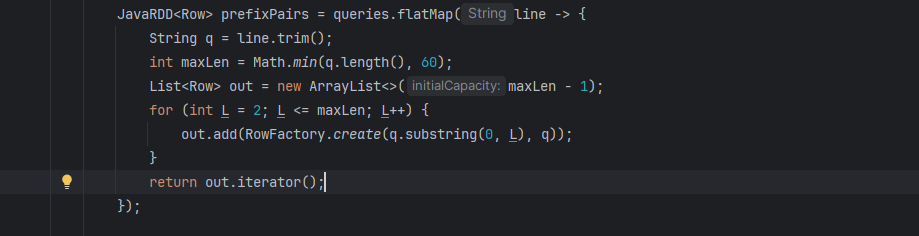
Step 0



STEP 1: Read queries from HDFS



STEP 2: Generate (prefix, fullQuery) pairs



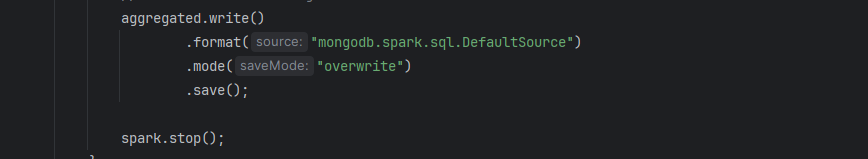
STEP 3: Count frequencies of each (prefix, query)



STEP 5: Assemble completions array with timestamp  
 Use current\_timestamp() for batch time



STEP 6: Write to MongoDB then stop spark



How to run

|  |
| --- |
| docker exec -it spark-master /bin/bash  /spark/bin/spark-submit --master spark://spark-master:7077 --packages org.mongodb.spark:mongo-spark-connector\_2.12:10.1.1 --class me.spark.TrendingAutocomplete /opt/spark-apps/spark-hdfs.jar hdfs://namenode:8020/logs/queries.txt  mymongo.autocomplete\_prefixes 5 |

Verify in mongodb :

|  |
| --- |
| docker exec -it mongodb mongosh  use mymongo  db.autocomplete\_prefixes.find().pretty() |