# Summary:

1. Used the SpringBoot framework for developing the task.
2. Used @Schedule annotation of SpringBoot for timing operations at an hourly frequency.
3. Used @Async annotation of SpringBoot for concurrent operations.

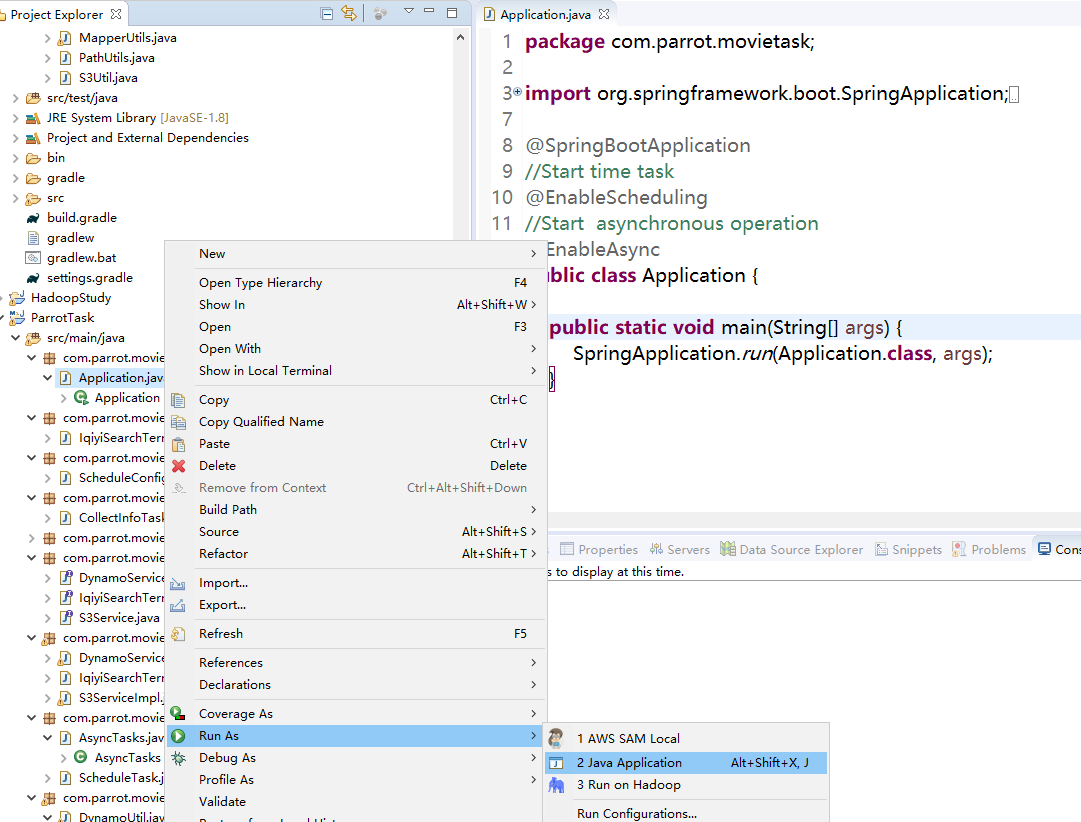
# Design:

In order to improve the collection efficiency, I paged MySQL table data. Customized the five methods, each of which performs a portion of the data collection operation, and asynchronously execute the five methods with @Async.

In order to split the raw dump Vimeo API JSON results into files of reasonable size optimized for apache spark reads, I stored the raw data in a local directory per hour, set **S3\_SPARK\_FILESIZE** in the configuration file, generate a small file by the split size, and upload it to S3. After the process, I used the URL returned by S3 to determine whether the last file had been successfully uploaded and deleted all files in the local directory.

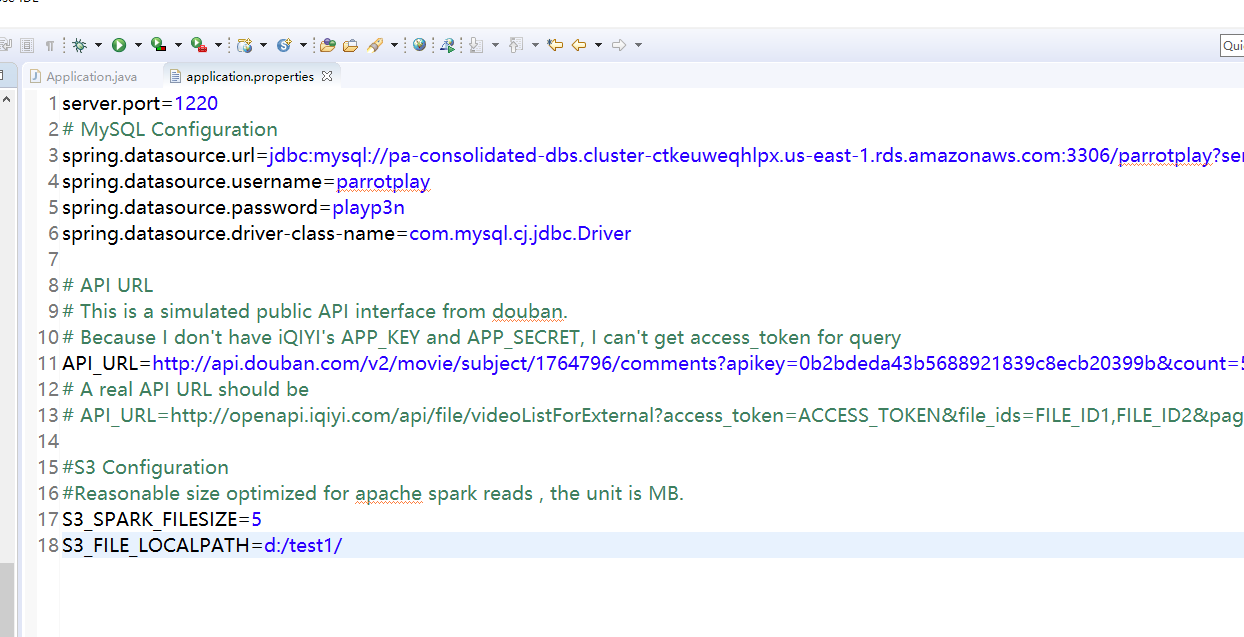
# Start:

Run com.parrot.movietask.Application.java in Java Application.



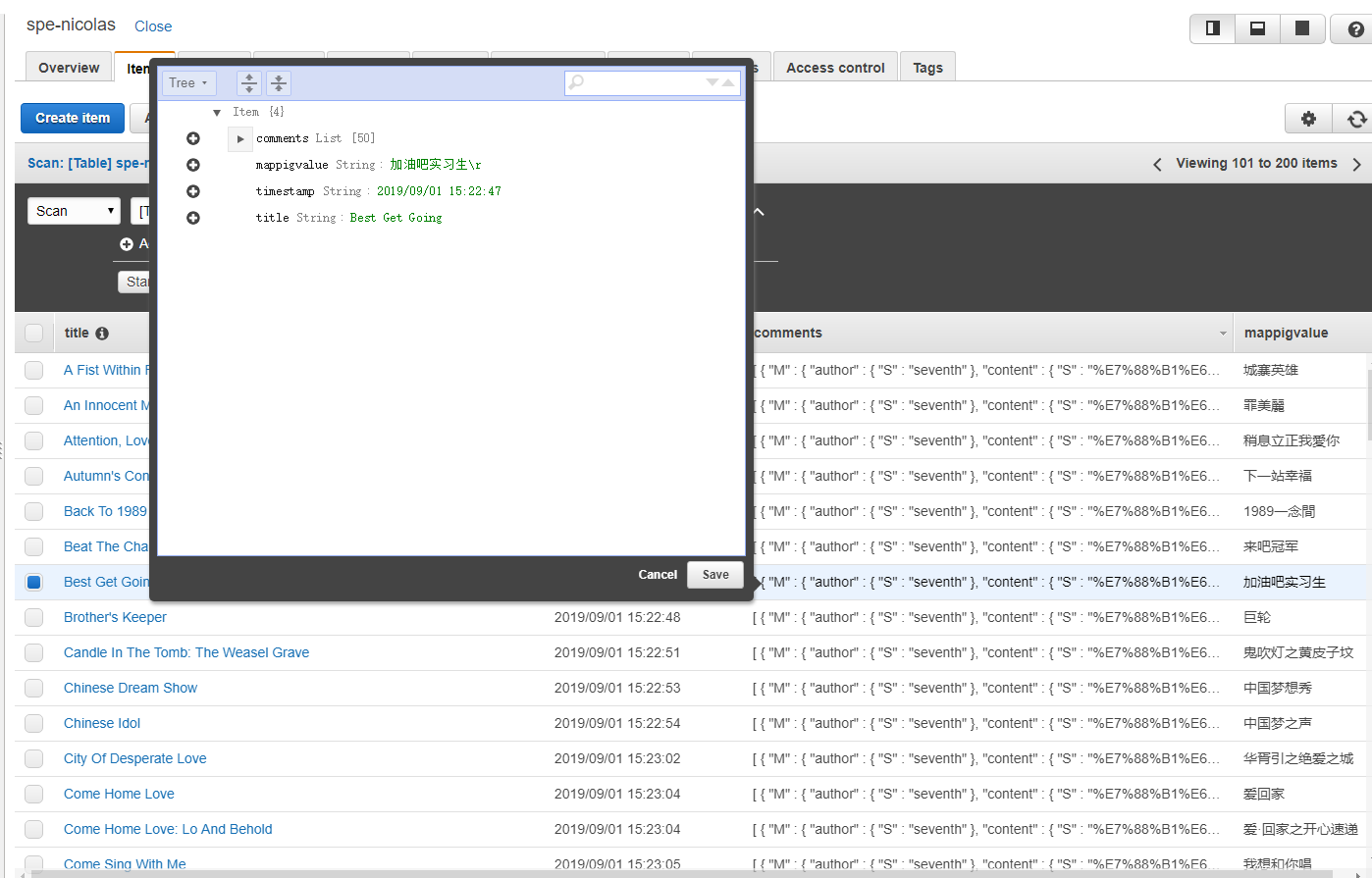
# Configuration:

The configuration file **application.properties** is in the directory of main/resources. It can be configured to suit your computer's development environment

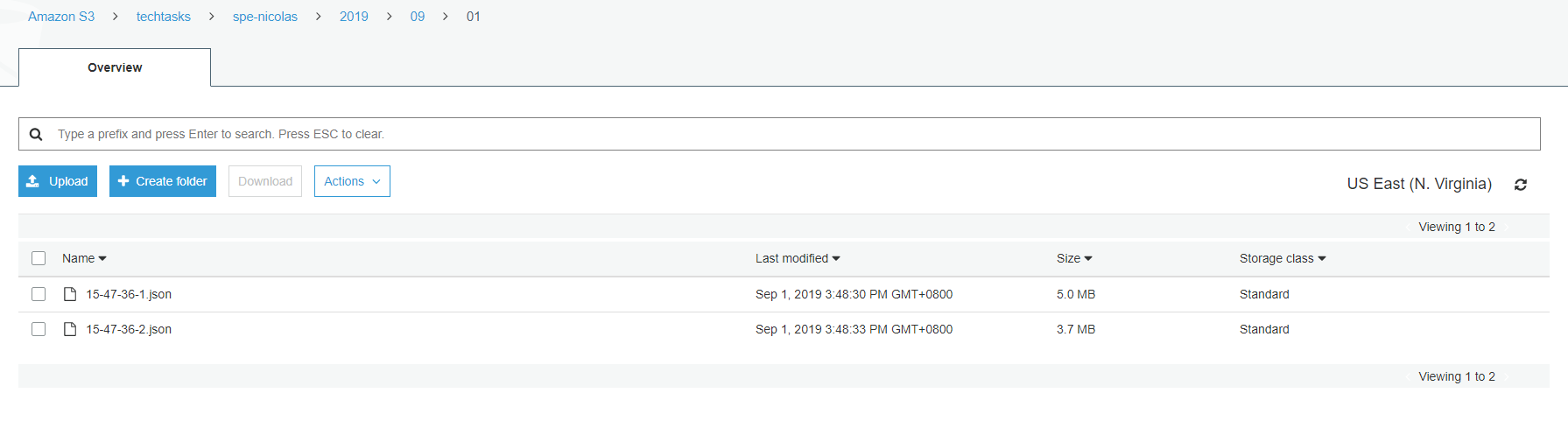


# Results

DynamoDB results

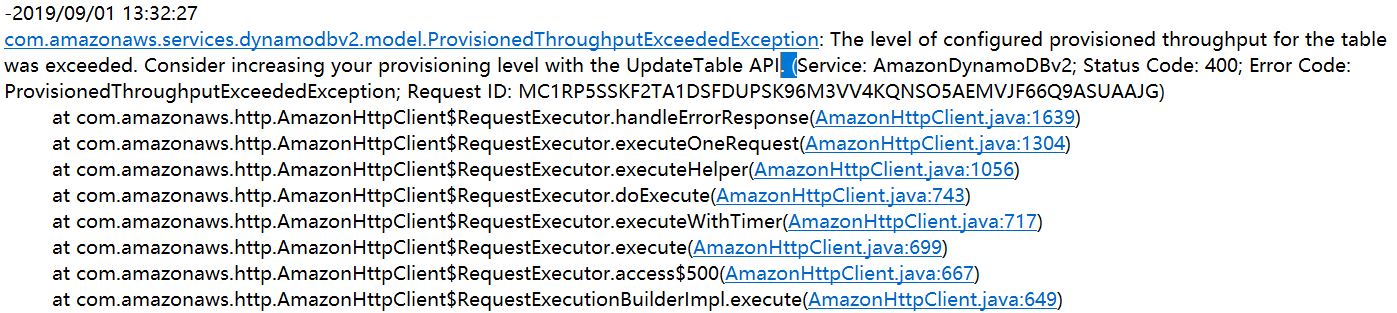


S3 results



# Issues:

1. Because I don’t have iqiyi API's app\_key and client\_secret, I can't get an access\_token through the API, so I can't call the API interface to get TV data. For this I used a public movie review interface from Douban. Although the data collected by each search term is same, as long as I can call the iQIYI interface, I can use the corresponding API to return the data and parse JSON into the AWS Dynamo database.
2. Due to AWS Dynamo's limited read-write capability, if the program is turned on @Async, it will report an error, but the single thread is fine.



# Improvement:

1. Considering the processing of 20k movies, 10 multi-threaded tasks can be customized to improve the collection efficiency.
2. Due to the time constraints, I used IO API to implement the large file read/write, split operations. In order to improve system performance, it is better to adopt NIO interfaces to implement these functions.
3. The method for determining whether all files are uploaded to S3 is not very accurate. Consider using schedule tasks to periodically delete locally stored files