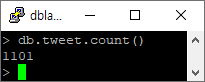
**데이터베이스시스템 프로젝트3**

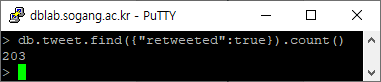
20171273 심현우

1. MongoDB 질의문
2. 전체 트윗의 개수를 구하시오.



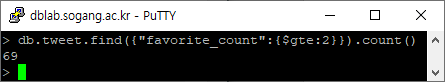
질의문 : db.tweet.count()

1. Retweet된 트윗의 개수를 구하시오.



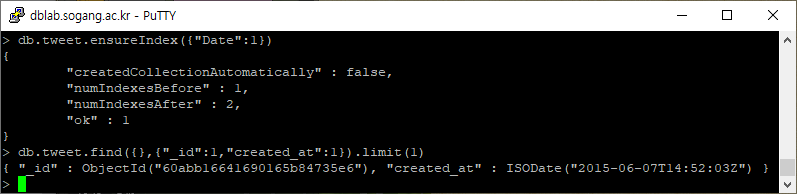
질의문 : db.tweet.find({“retweeted”:true}).count()

1. 좋아요(favorite\_count)가 2 이상인 트윗의 개수를 구하시오.



질의문 : db.tweet.find({“favorite\_count”:{$gte:2}}).count()

1. 전체 트윗을 날짜 별로 정렬해서 가장 빠른 날짜의 트윗을 출력하시오. (ObjectID와 날짜만 출력할 것.)

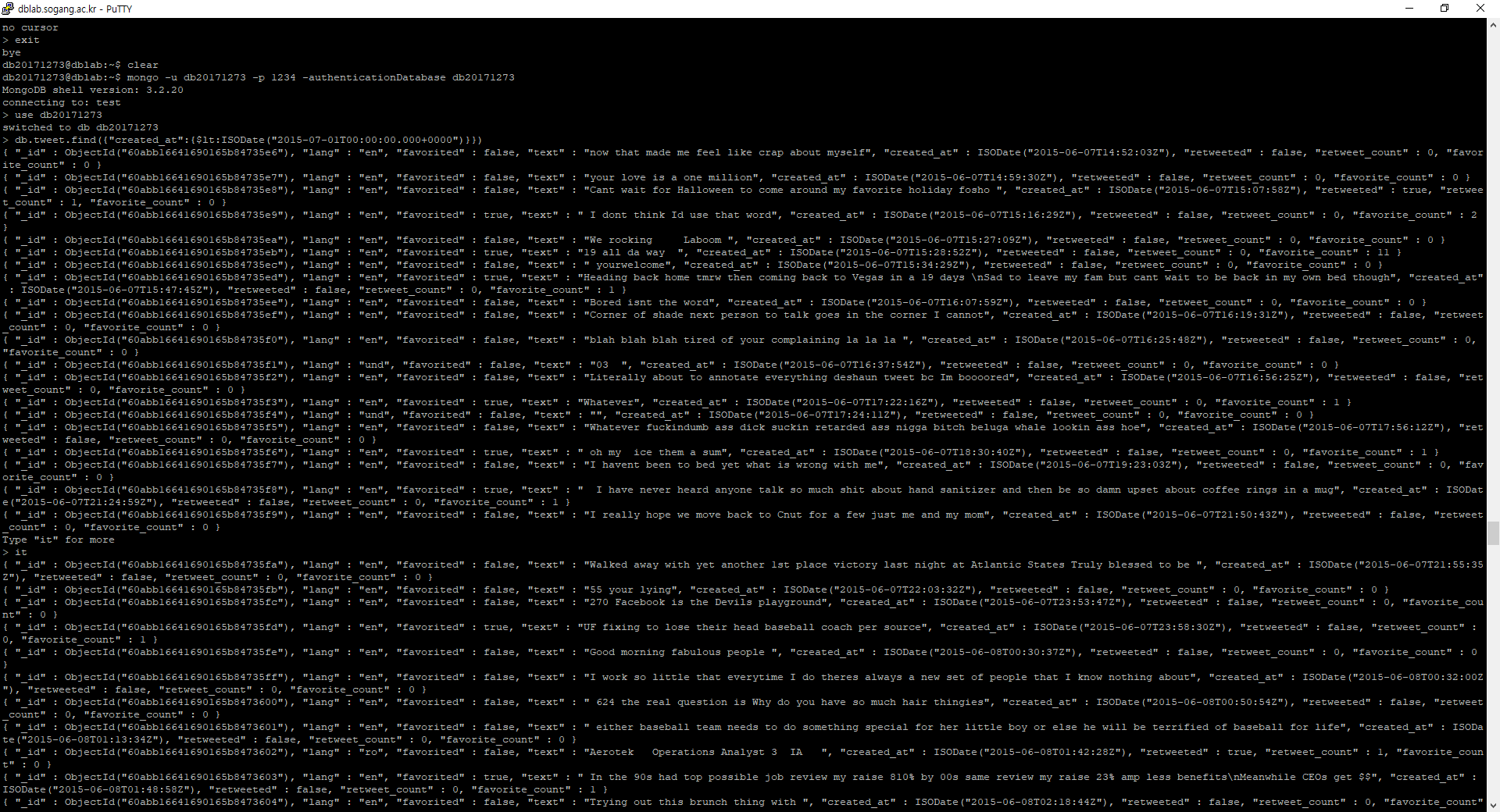


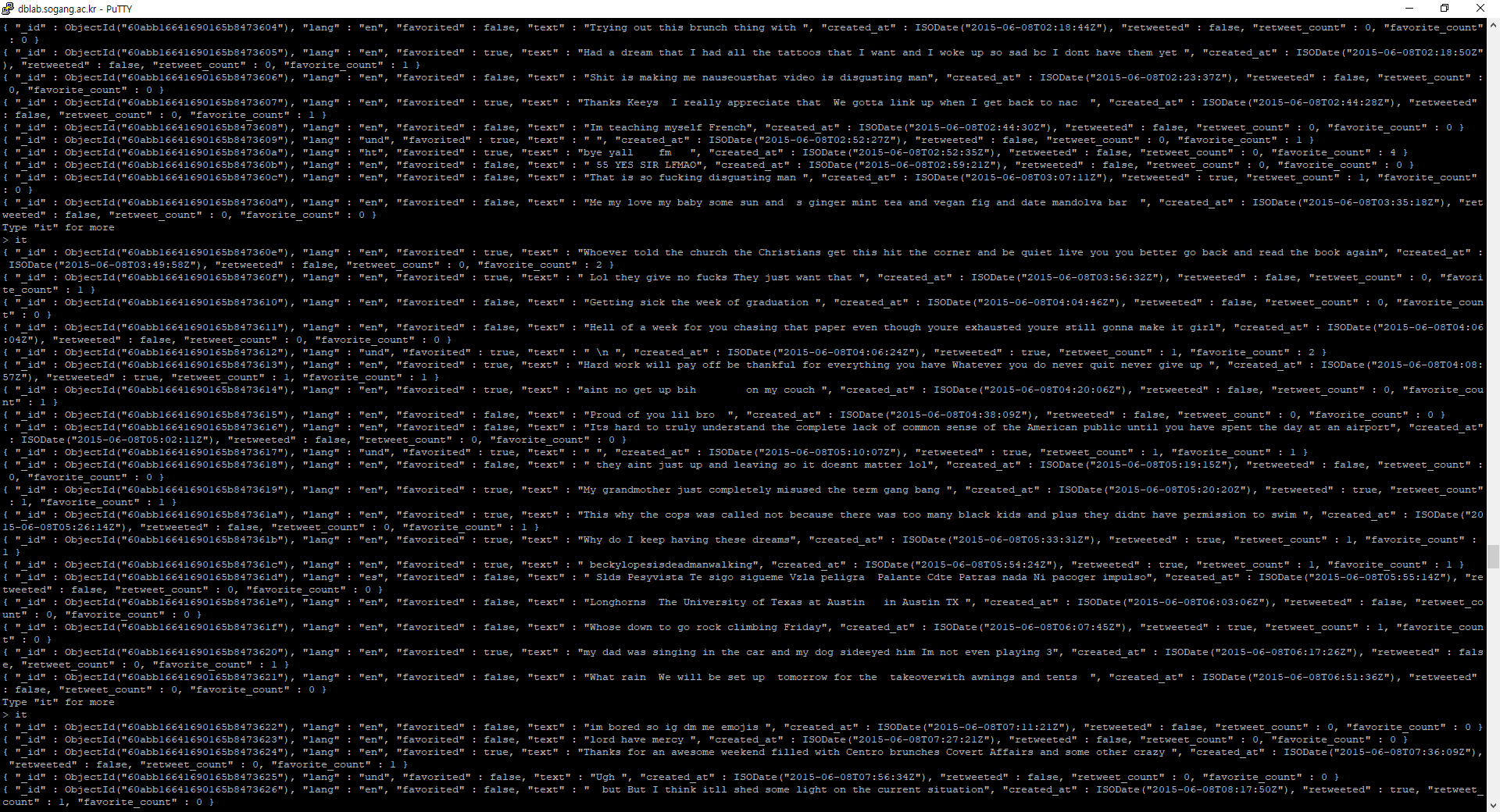
질의문 : db.tweet.ensureIndex({“Date”:1})

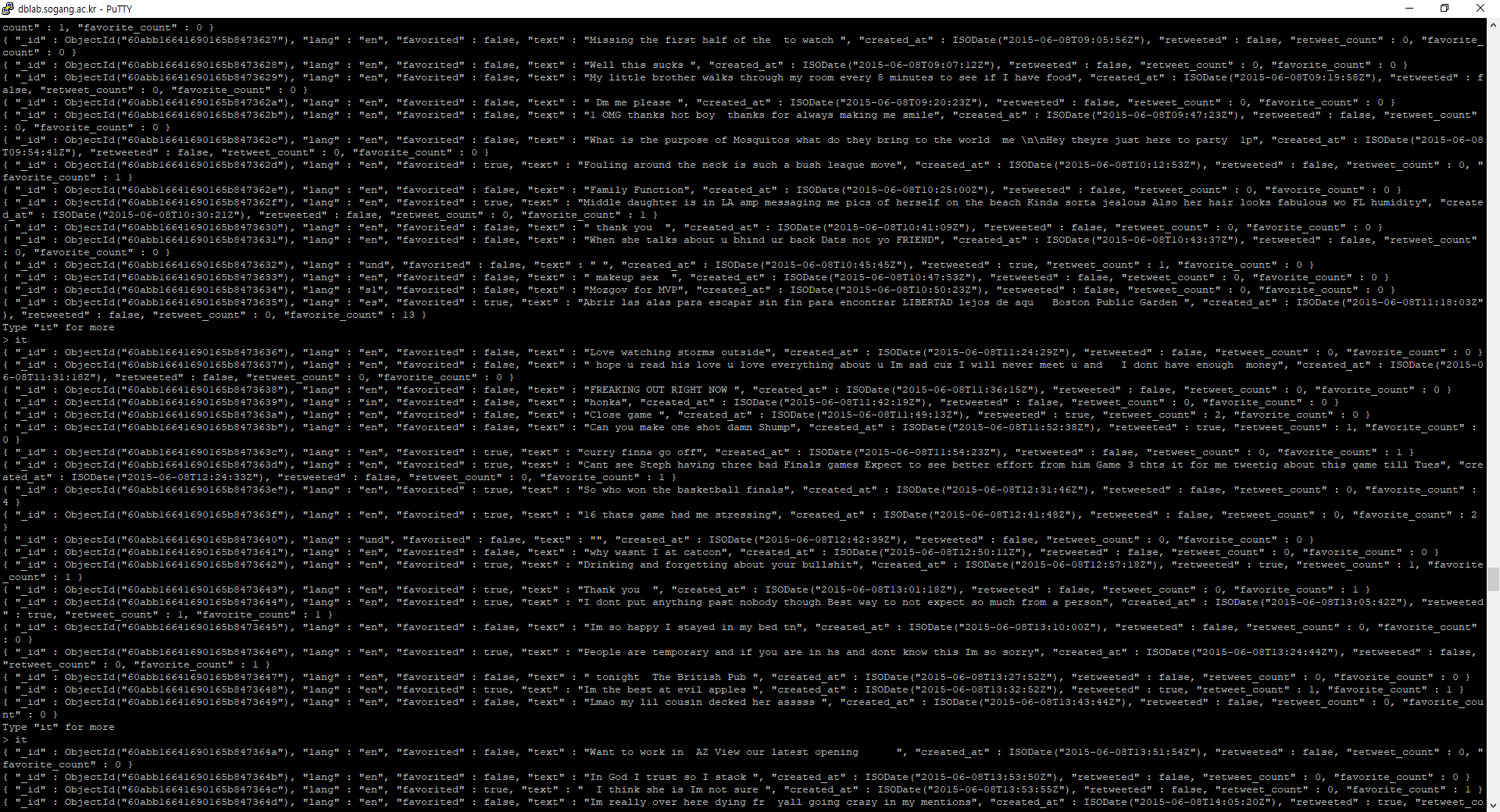
Db.tweet.find({},{“\_id”:1,”created\_at”:1}).limit(1)

우선 index를 통해 date를 오름차순으로 트윗 데이터를 정렬하고, find에서 id와 date만 나올 수 있도록 select한 후에 limit(1)로 첫 데이터만 출력하도록 한다.

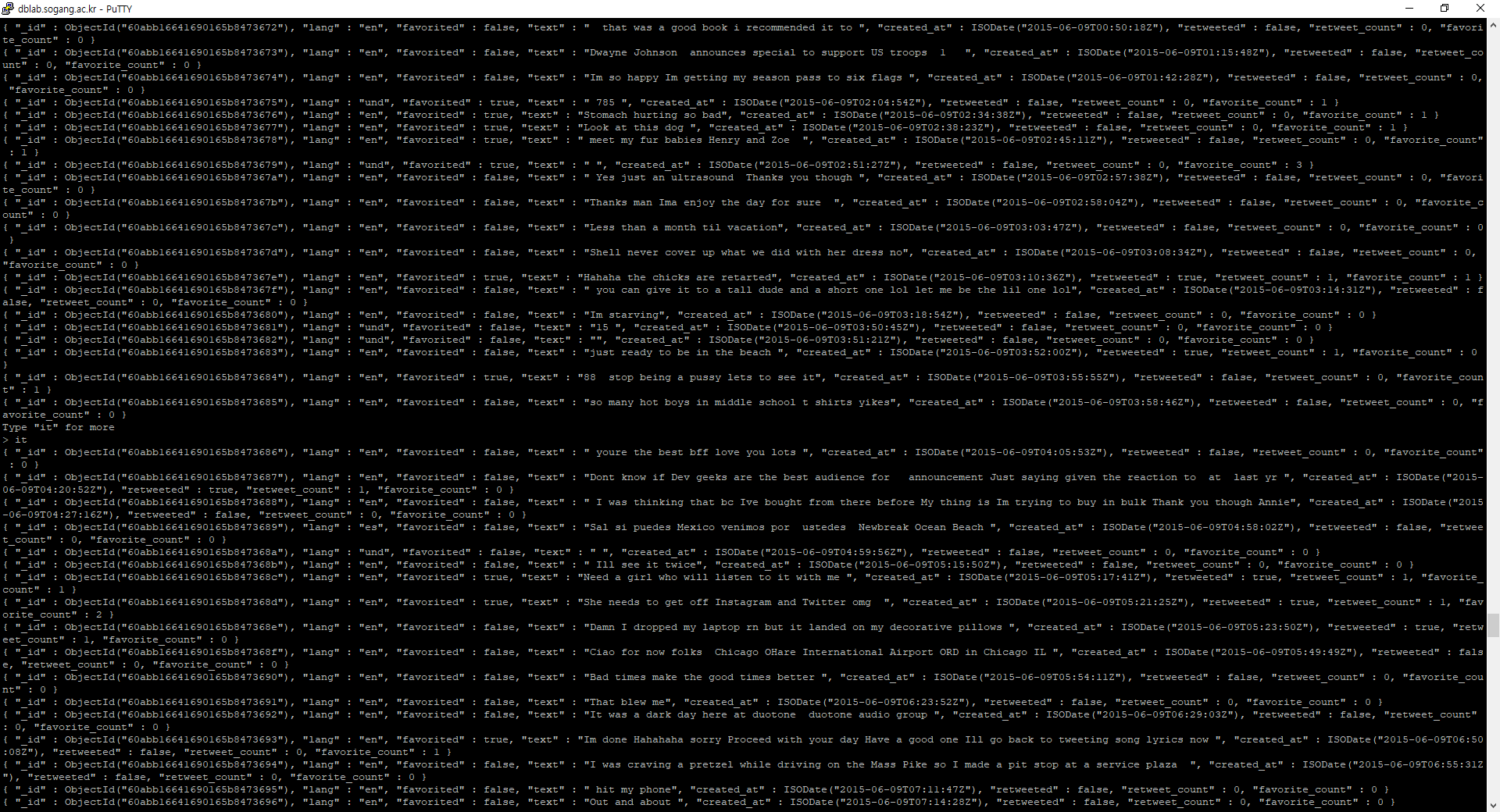
1. 2015년 6월 1일부터 2015년 6월 30일까지 6월 한 달 동안 실린 기사의 개수를 구하시오. (6월 30일에 올라온 트윗도 포함.)

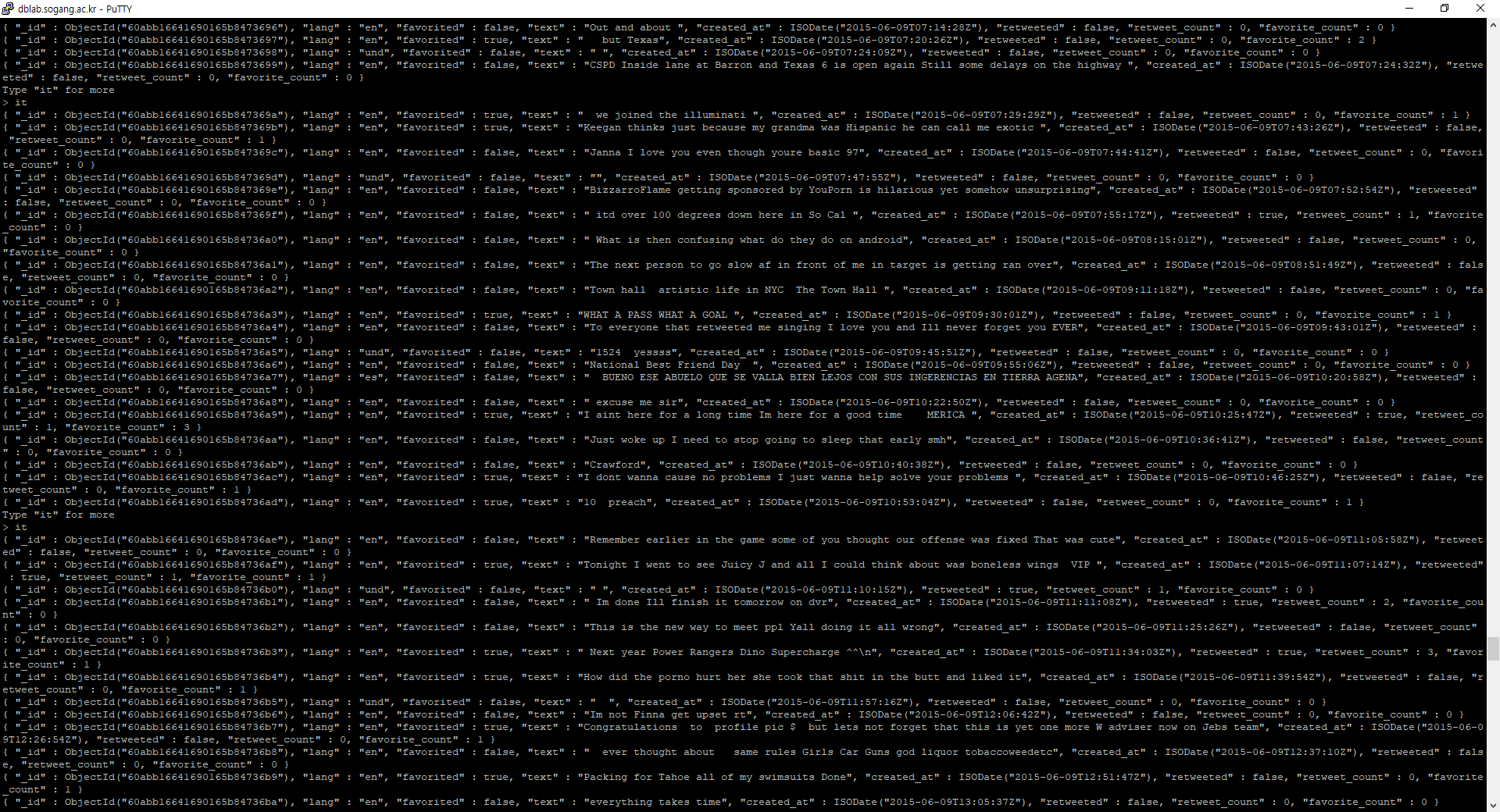


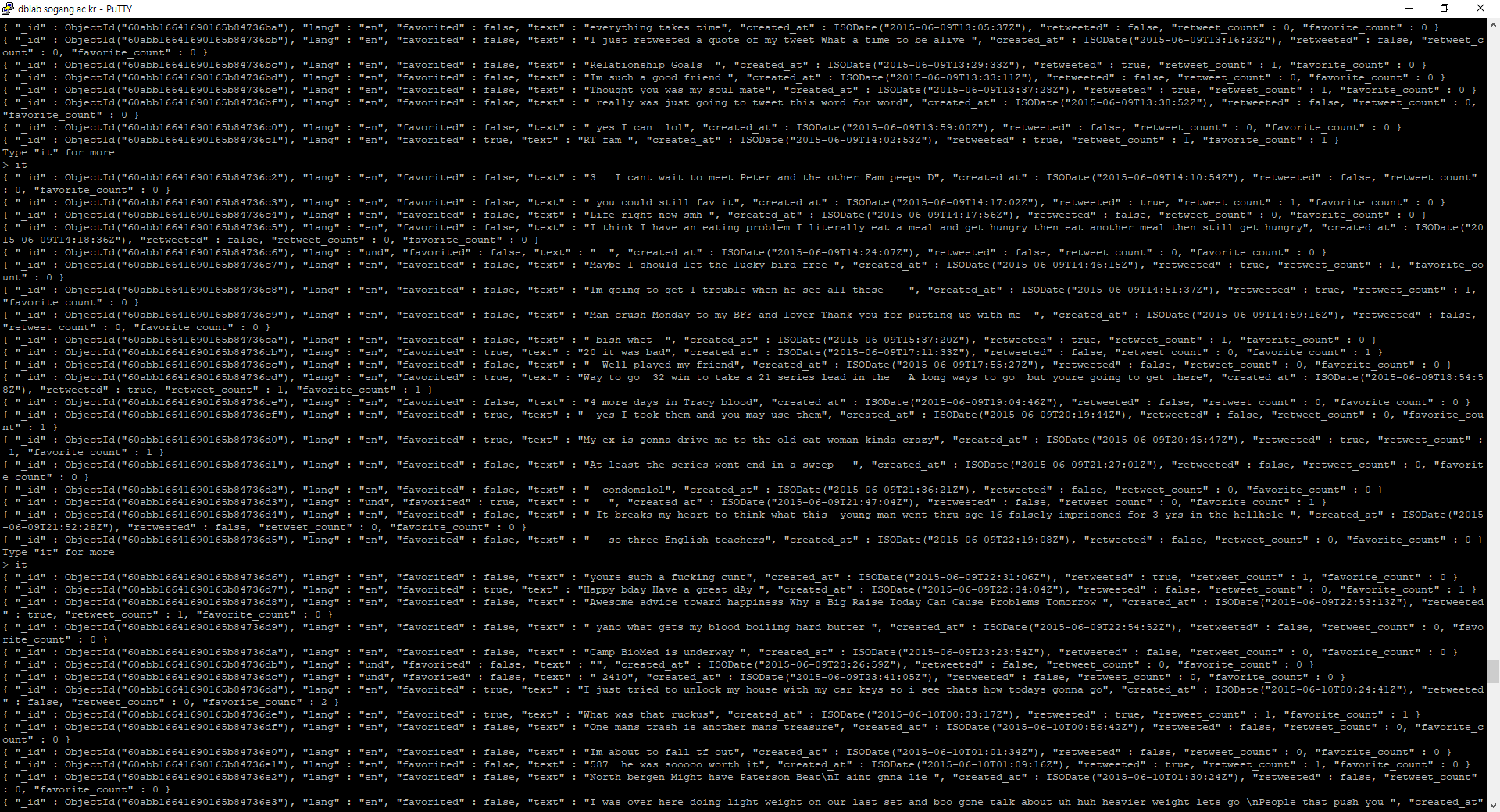


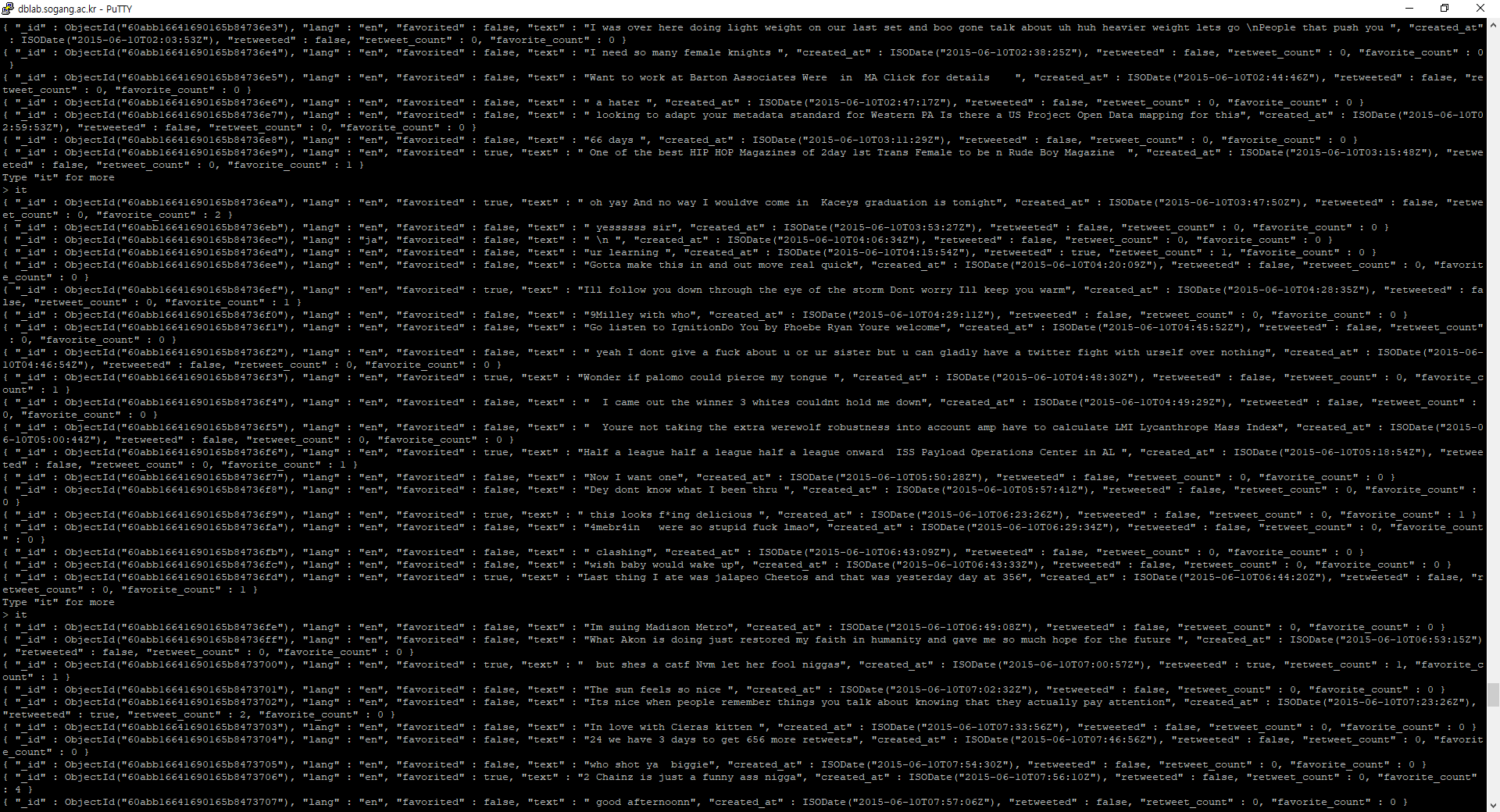


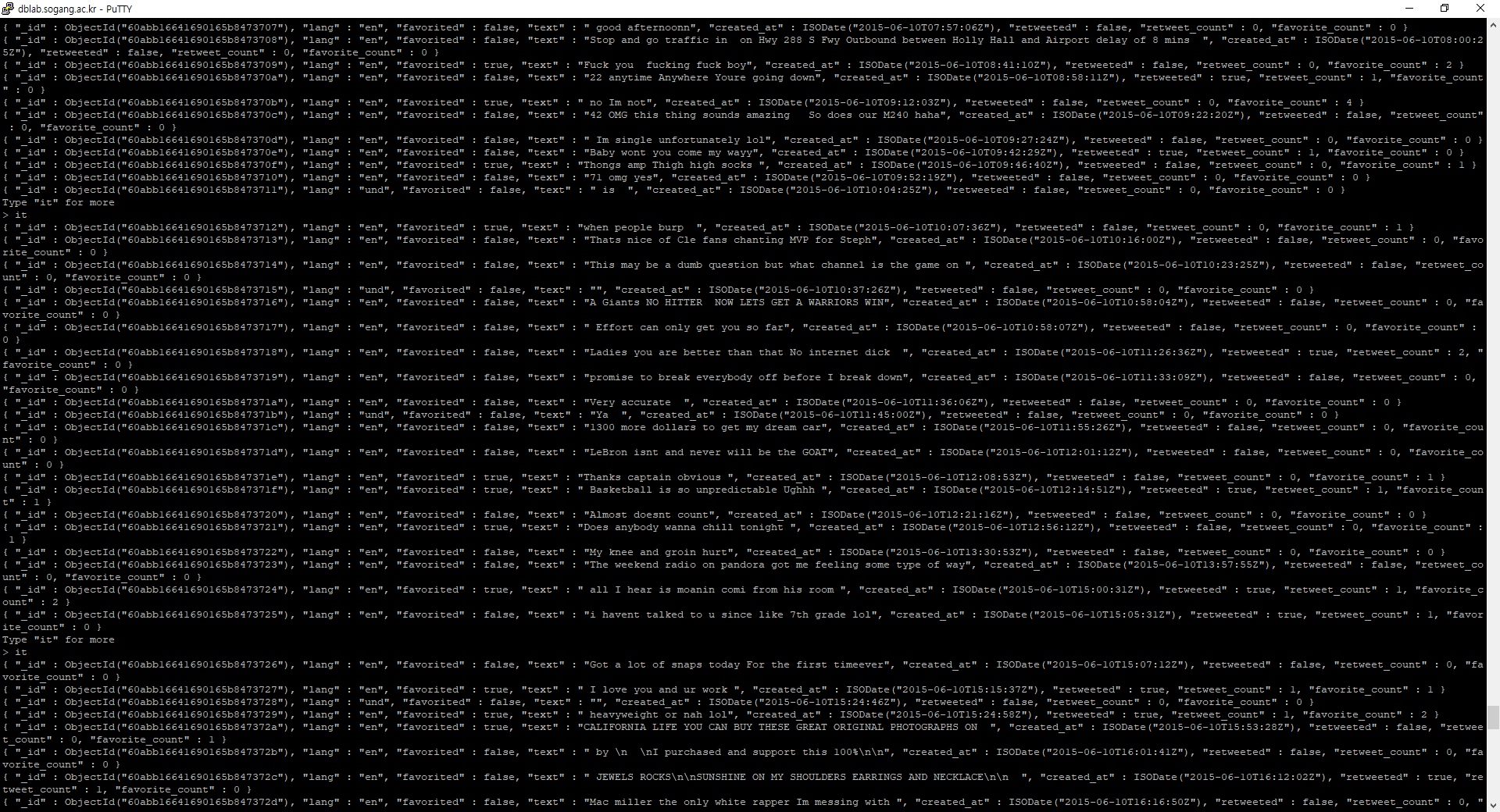


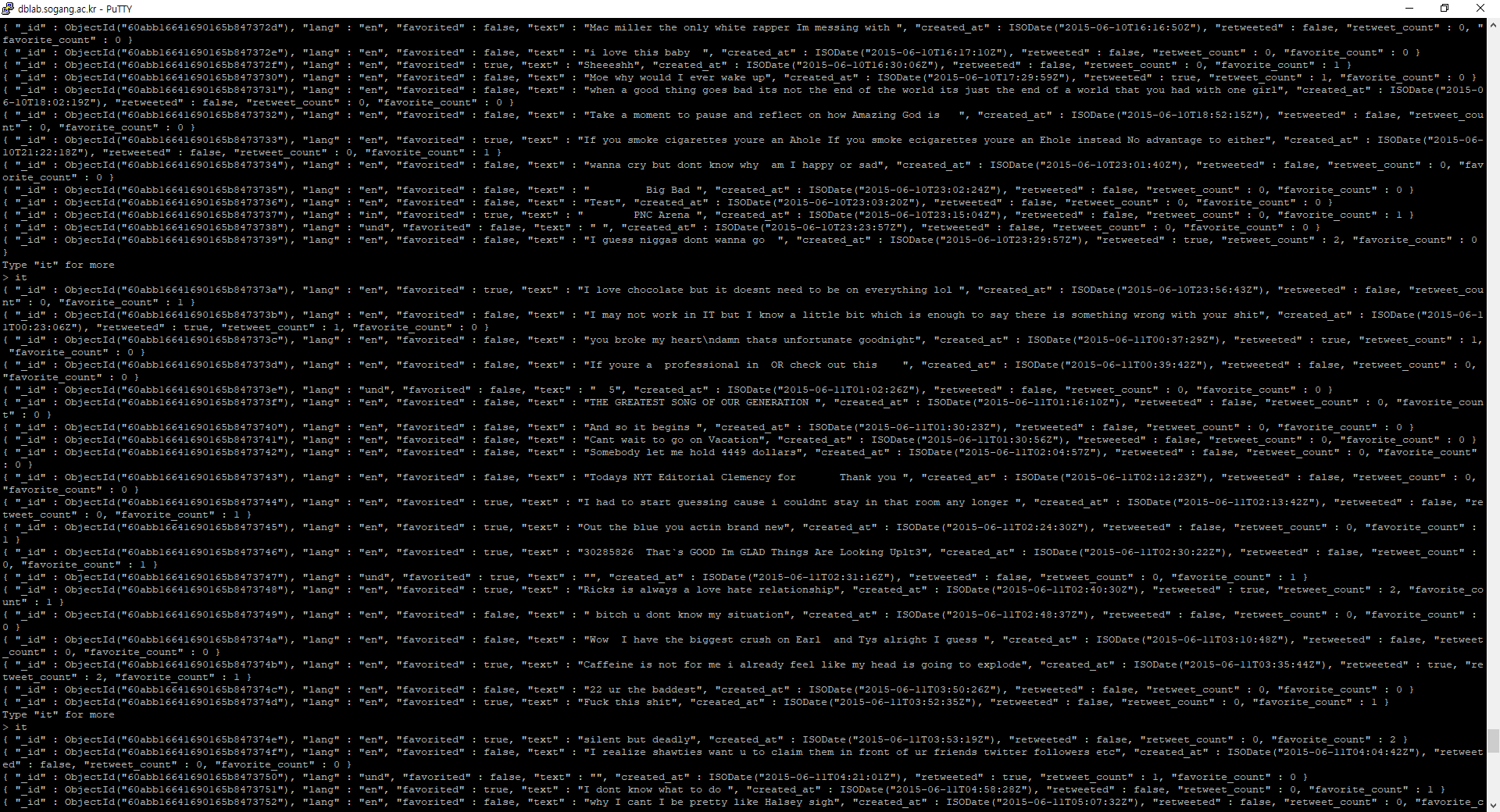


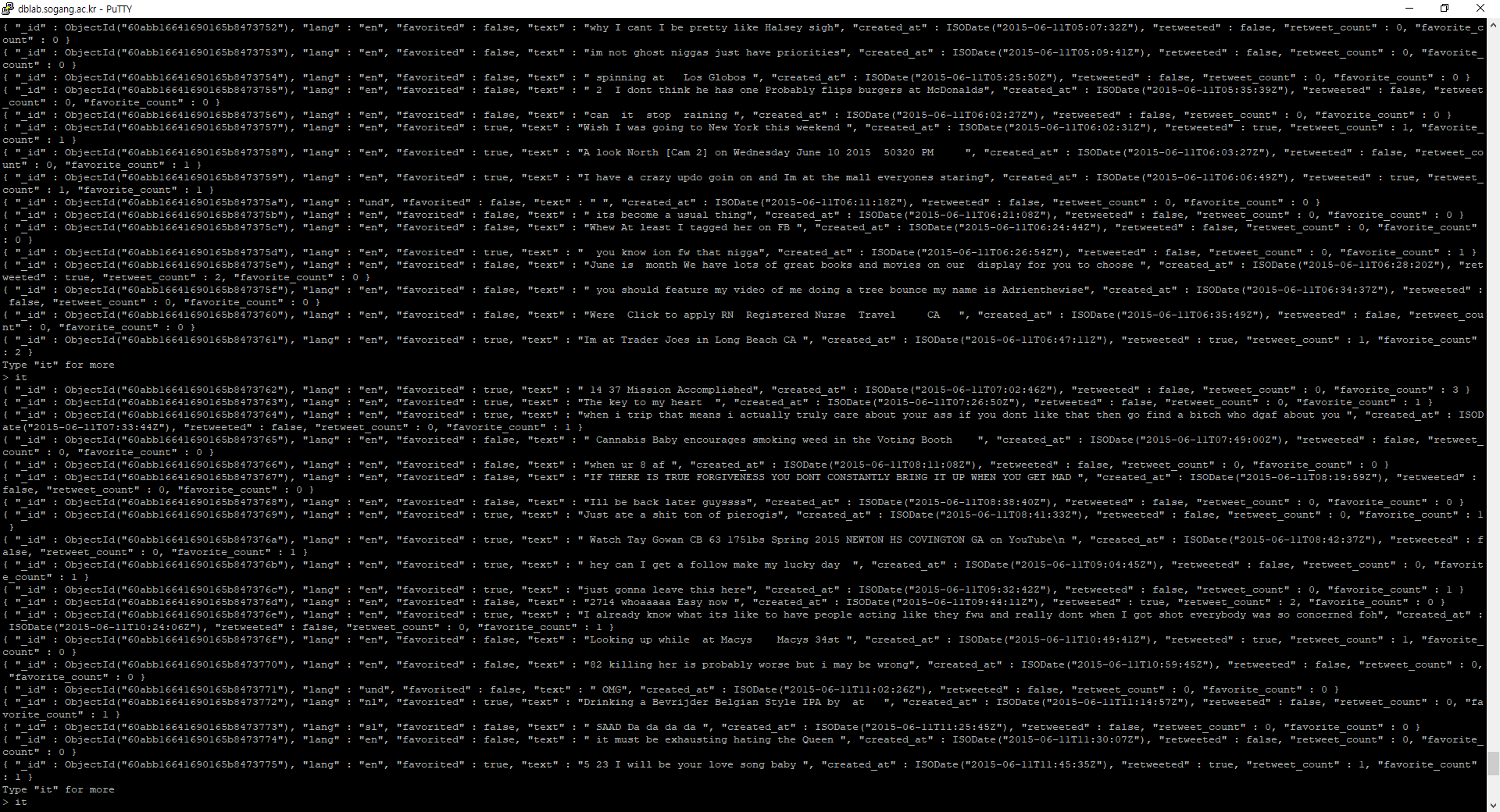


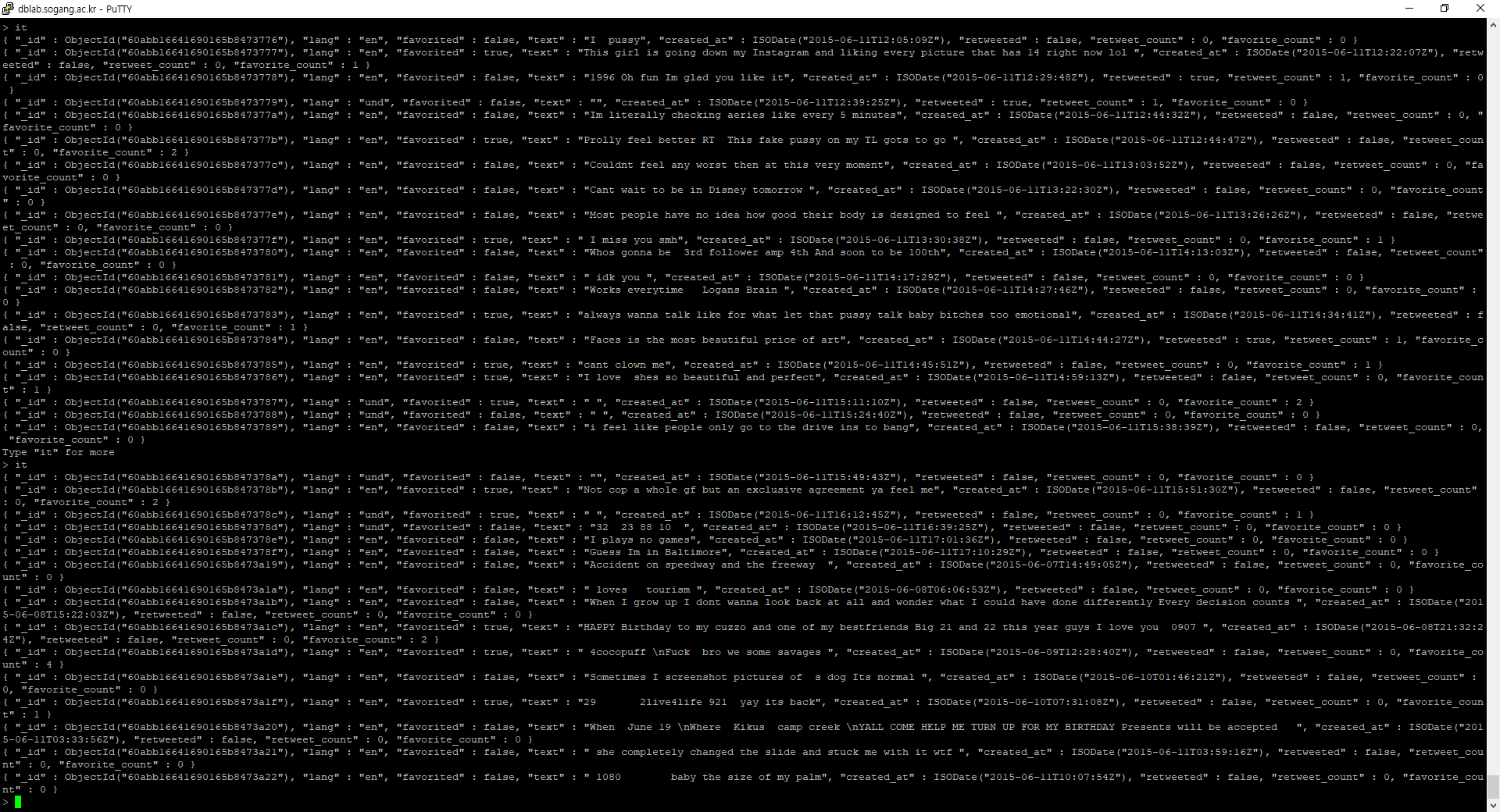












질의 : db.tweet.find({“created\_at”:{$lt:ISODate(“2015-07-01T00:00:00.000+0000”)}})

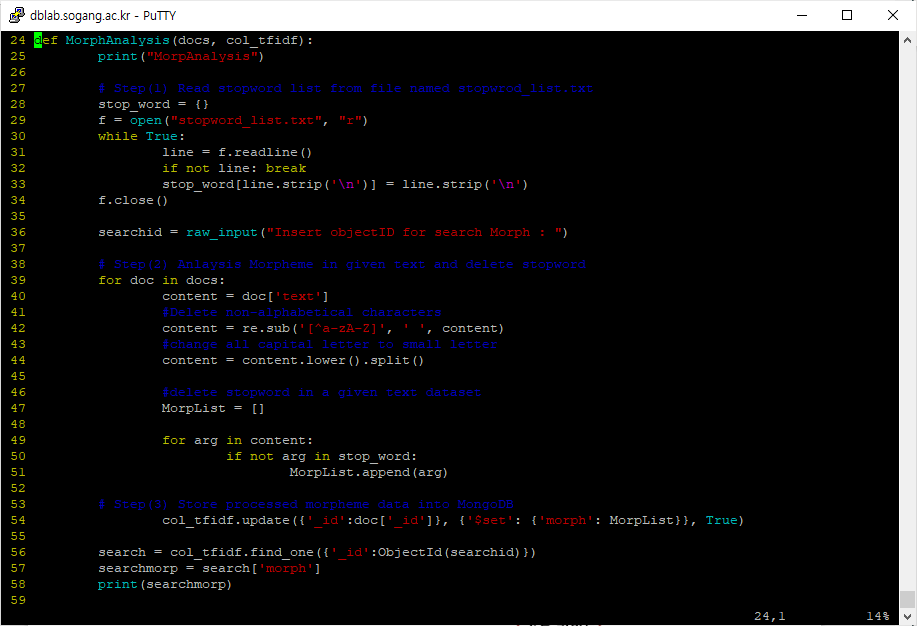
1. RDB VS. NoSQL DB

RDB는 사용자의 요구에 따라 정보를 생성하고 데이터를 관리할 수 있는 시스템이다. RDB에서의 저장 방식은 SQL에 따르고 있으며 정해진 스키마에 따라 저장해야 한다. 또한 RDB의 R은 relation으로 관계형 데이터베이스, 즉 key에 따라 관계를 가지고 있는 것이 특징이다. 따라서 이 RDB의 장점은 정해진 스키마에 따라 명확한 구조를 가지고 있다는 점이다. 하지만 단점으로는 데이터의 양이 많아질수록 데이터 간 관계를 맺어야 하기 때문에 복잡한 구조가 될 수 있다는 것과, 성능 향상을 위해서는 서버의 성능을 향상 시켜야만 한다는 것이다.

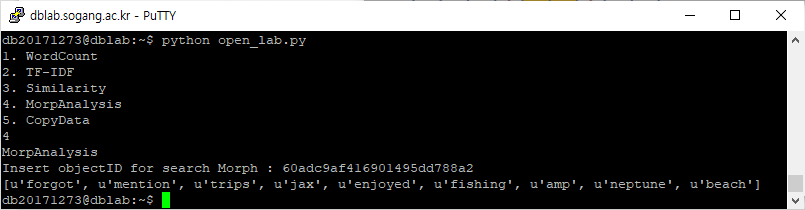
반면 NoSQL은 Not only SQL의 줄임말로 RDB와는 다르게 관계형 데이터베이스를 가지고 있지 않다는 것이다. 따라서 데이터 간 관계를 가지지 않아도 되며 데이터가 많아져도 관리가 용이하다는 장점이 있다. 또한 성능을 향상시키기 위해서 데이터를 분산 저장하는 scale-out을 할 수 있다. 스키마가 없기 때문에 유연하며 자유로운 데이터 구조를 가질 수 있다. 하지만 데이터의 중복이 발생할 수 있다는 것과, 스키마가 없기에 명확한 데이터 구조가 없다는 것이 단점이다.

텍스트 마이닝을 할 때는 NoSQL이 더 유용하게 사용될 수 있다. 우선 이번 프로젝트에서 다루는 tweet은 전세계적인 SNS이기 때문에 방대한 양의 데이터가 있다. 많은 양의 데이터를 처리하는데 유용한 NoSQL을 이용하는 것이 좋다. 또한 텍스트 마이닝 자체에서는 데이터 간 관계에 주목하는 것이 아니기 때문에 정해진 스키마를 통해 데이터 구조를 만들어야 할 필요성이 없다. 따라서 빠르고 scale\_out을 이용할 수 있는 NoSQL을 이용하는 것이 더 유용하다.

1. python코드
2. 형태소 분석 및 불용어 처리

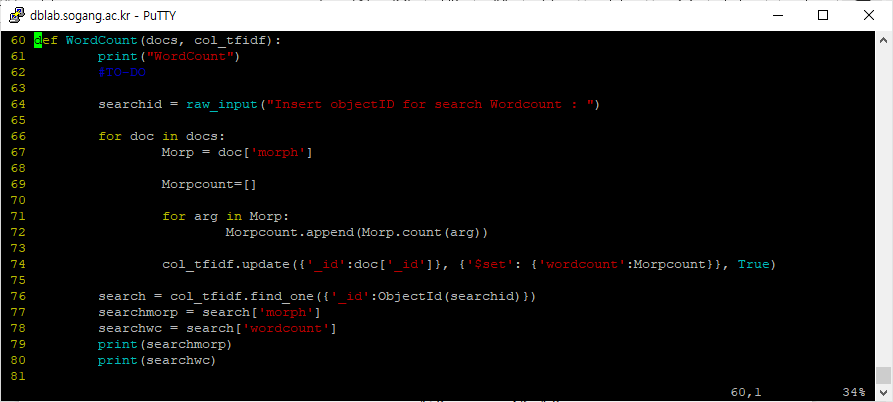


기존 코드에 raw\_input을 통해 search id 즉 찾을 아이디를 입력받는다. 입력받은 아이디에 맞게 find\_one과 이미 만들어진 morph를 통해 형태소를 출력한다.

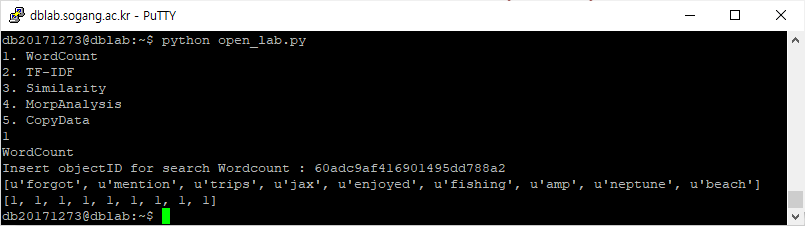


프로그램 실행 결과는 다음과 같다.

1. Word count 구하기

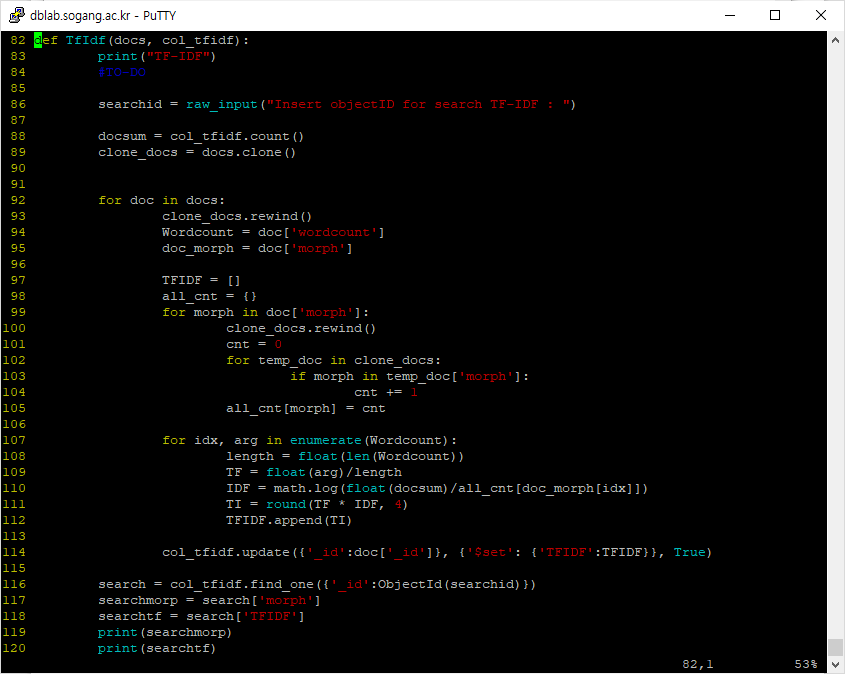


Word count 수행 코드는 다음과 같다. 각 doc를 for문을 통해 접근하고 morphcount 배열을 만들어 morph배열의 단어들을 돌면서 단어의 빈도를 count를 통해 구하여 morphcount배열에 할당한다. 이 후 입력받은 아이디에 맞게 morph와 morphcount를 출력한다.

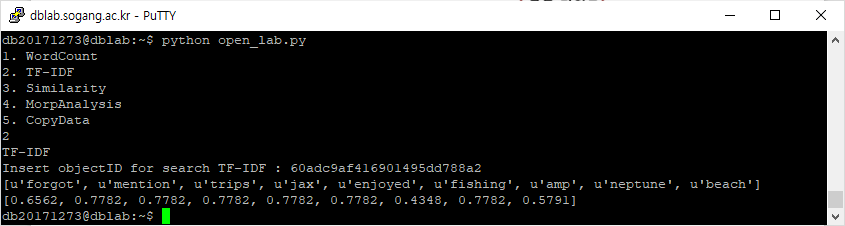


프로그램 수행 결과는 다음과 같다.

1. TF-IDF 수행

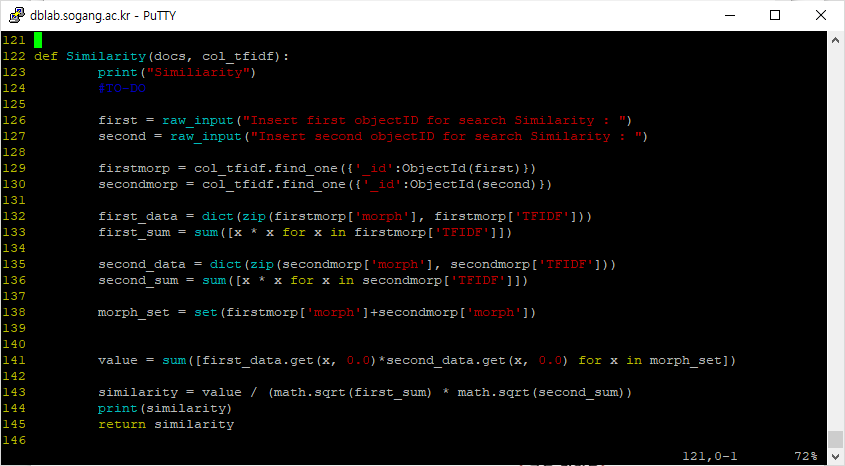


명세서의 tf\_idf구하는 방법을 이용하여 각 doc의 morph에 접근하여 각 doc의 morph 형태소에 대한 전체 doc에서의 빈도수를 구한다. 이후 구하는 공식에 맞게 len와 math의 log함수를 이용하여 tf-idf값을 구하고 TFIDF배열에 할당하여 mongodb에 추가한다.

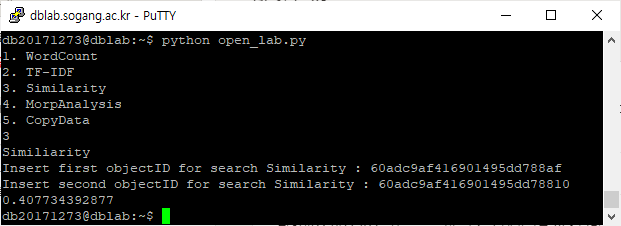


프로그램 수행 결과는 다음과 같다.

1. 문서 유사도 구하기



유사도 구하는 방법은 다음과 같다. 우선 두가지 아이디를 받은 후 아이디에 대한 값들에 접근한다. 이후 dict와 zip을 이용하여 각각의 아이디에 대한 morph와 tf-idf 가중치를 매칭한다. 매칭하고 유사도 구할 때 분모에 쓰일 가중치의 제곱의 합을 구한다. Set을 이용하여 두 아이디에 대한 합집합을 구하고 가중치의 벡터의 곱을 get을 이용하여 없는 단어에 대한 가중치는 0으로 만들어 계산을 진행한다. 이후 마지막 유사도를 구하기 위해 value값에서 sum에 루트를 math.sqrt로 완성한다.



프로그램 수행 결과는 다음과 같다.