Assignment

DS^2 과정 (MongoDB)

Fall 2018

주의사항

제출기한 : 2018.11.14. 11시 59분 59초
참여 인원 : 3인 1조
파일 다운로드 : ds2board2/assignment
제출파일 : 아래와 같은 파일을 압축하여 제출 (zip, gz)

1. python code [notebook 코드(확장자 .ipynb)는 제출 불가].
2. 프로그램 설명 문서 pdf파일 (실행과정 및 MongoDB 함수 등)

프로그램 실행 방법은 결과 화면과 동일하다고 하며, 변경이 필요한 경우 구현 환경에 관한 명세 필수.

제출장소 : ds2homework@gmail.com
문의사항 : koo@dbs.snu.ac.kr, stdio@dbs.snu.ac.kr

늦게 제출할 경우 2018.11.15. 11시 59분 59초까지 받을 예정이며, panalty로 10%가 있음. 그 이후에는 제출하실 수 없으며 (0점 처리), 표절도 0점 처리됨.

코드 제출 전 불 이익을 받지 않도록, 코드가 제대로 실행이 되는지 확인하시고 제출하시기 바랍니다.

1 Grading

grades.json을 import하여 학생들의 성적을 매기려고 한다. 성적은 quiz와 exam, homework로 구성되고, 학생들의 ID (sid)는 0부터 99까지이다. 아래와 같은 조건을 PyMongo를 사용하여 grade.py에 A-C를 구현하시오.
database는 ds2, collection 이름은 grades이며 grades.json은 import가 되어 있다고 가정한다.

A) Figure 1과 같이 학생성적을 10개를 출력하는 pagination 함수를 만드시오. sid 순서대로 grades와 sid만 출력한다.

```
$ python3 grading.py 1 2
{'sid': 10, 'grades': [{'score': 46, 'type': 'quiz'}, {'score': 81, 'type': 'exam'}, {'score': 39, 'type': 'homework'}], 'note': 'great'}
{'sid': 11, 'grades': [{'score': 11, 'type': 'homework'}, {'score': 92, 'type': 'exam'}, {'score': 5, 'type': 'quiz'}]}
{'sid': 12, 'grades': [{'score': 47, 'type': 'homework'}, {'score': 79, 'type': 'quiz'}, {'score': 51, 'type': 'exam'}]}
{'sid': 13, 'grades': [{'score': 31, 'type': 'homework'}, {'score': 66, 'type': 'quiz'}, {'score': 70, 'type': 'exam'}]}
{'sid': 14, 'grades': [{'score': 53, 'type': 'homework'}, {'score': 9, 'type': 'quiz'}, {'score': 84, 'type': 'exam'}]}
{'sid': 15, 'grades': [{'score': 78, 'type': 'homework'}, {'score': 75, 'type': 'exam'}, {'score': 31, 'type': 'quiz'}]}
{'sid': 16, 'grades': [{'score': 81, 'type': 'homework'}, {'score': 97, 'type': 'exam'}, {'score': 80, 'type': 'quiz'}], 'note': 'good answers'}
{'sid': 17, 'grades': [{'score': 54, 'type': 'homework'}, {'score': 92, 'type': 'exam'}, {'score': 48, 'type': 'quiz'}]}
{'sid': 18, 'grades': [{'score': 61, 'type': 'homework'}, {'score': 66, 'type': 'exam'}, {'score': 80, 'type': 'quiz'}]}
{'sid': 19, 'grades': [{'score': 44, 'type': 'quiz'}, {'score': 81, 'type': 'homework'}, {'score': 66, 'type': 'exam'}]}
```

Figure 1: command - python3 grading.py 1 2

B) 아래와 같은 식을 사용하여 Figure 2과 같이 학생들의 성적등급을 결정하는 letter 함수를 작성하시오. 아래와 같은 방법으로 총점을 계산하여 학생 성적의 최종 등급을 결정하여, 각 학생의 총점 및 등급을 추가하시오 (total, letter 필드 생성). 그 후 총점이 높은 순서대로 sid, total, letter만 출력해 보시오.

```
$ python3 grading.py 2
{'sid': 54, 'total': 91.2, 'letter': 'A'}
{'sid': 16, 'total': 88.8, 'letter': 'B'}
{'sid': 5, 'total': 87.5, 'letter': 'B'}
{'sid': 99, 'total': 87.1, 'letter': 'B'}
{'sid': 72, 'total': 86.6, 'letter': 'B'}
{'sid': 93, 'total': 85.4, 'letter': 'B'}
{'sid': 63, 'total': 84.9, 'letter': 'B'}
{'sid': 23, 'total': 83.2, 'letter': 'B'}
{'sid': 44, 'total': 82.9, 'letter': 'B'}
{'sid': 52, 'total': 82.6, 'letter': 'B'}
```

Figure 2: command - python3 grading.py 2 (문서의 일부분 출력)

- Total score = quiz × 0.2 + homework × 0.3 + exam × 0.5
- 성적 등급 기준:
  >=90: A, >=80: B, >=70: C, >=60: D, 나머지: F

C) 수업시간에 발표를 잘한 학생에게 가산점을 주기 위해 note필드에 메모를 해두었다. 학생들 중 exam이 100이거나 메모가 있는 학생들에게 총점에 가산점 10점을 더 주기로 하여 성적을 부여한다. 아래의 기준으로 Figure 3과 같이 relative collection에 각 학생 등급에 맞는 문서를 만들고, relative collection에서 sid 순서대로 sid, letter만 출력해 보시오.

```
$ python3 grading.py 3
{'sid': 0, 'letter': 'B'}
{'sid': 1, 'letter': 'D'}
{'sid': 2, 'letter': 'C'}
{'sid': 3, 'letter': 'B'}
{'sid': 4, 'letter': 'B'}
{'sid': 5, 'letter': 'A'}
{'sid': 6, 'letter': 'B'}
{'sid': 7, 'letter': 'C'}
{'sid': 8, 'letter': 'B'}
{'sid': 9, 'letter': 'C'}
{'sid': 10, 'letter': 'B'}
```

Figure 3: command - python3 grading.py 3 (문서의 일부분 출력)

- Total score = quiz × 0.2 + homework × 0.3 + exam × 0.5
- 만약 note field가 있거나 exam이 100점일 경우 총점 +10점을 준다. 둘다 있더라도 +10점만 부여한다. 총점은 100점을 넘지 못한다.

- relative collection 문서는 sid, total, letter field를 가진다.(_id는 자동 생성)
- 성적 등급 기준 : x의 식과 분포에 따른 성적은 다음과 같다.
  x= (total-min)/(max-min) * 100
  x >=80% : A, 80 > x >=50% : B, 50 > x >=20% : C, 20 > x >=10% : D, 나머지 : F

2 Pokemon Go

PyMongo를 사용하여 pokedex.json에서 질의문을 사용하여 정보를 다루고자 한다. 이 파일은 1세대 포켓몬 151종에 관한 정보를 가지고 있다. database는 ds2, collection 이름은 pokedex을 사용한다. pokedex.json은 import가 되어 있다고 가정한다.

A) 주인공 지우는 최고의 포켓몬 트레이너가 되기 위해 라이벌인 오바람을 이겨야 한다. 오바람은 이름이 Scyther과 Vileplume, Butterfree인 포켓몬들을 가지고 있다. 지우는 포켓몬이 없어 야생포켓몬을 잡아야 하는데, 20:00 ~ 24:00 사이에 출몰하는 포켓몬을 찾으려고 한다.
오바람이 가지고 있는 포켓몬들의 공통된 약점 속성(weaknesses 필드) 알아낸다. 이들 중 최소한 하나 이상의 속성(type 필드)을 가지고, 특정 시간대에 출몰(spawn_time 필드)하는 포켓몬을 찾아본다. Figure 4과 같이, 결과는 name으로 오름차순 정렬한 후, id와 name, spawn_time, type에 해당하는 값을 출력하시오.
   - 시간의 경우 String 이기 때문에 비교 연산 시 주의한다.
   - 공통 약점 속성 예시 : Caterpie = {Fire, Flying, Rock} ∩ Bulbasaur = {Fire, Ice, Flying, Psychic}
     Caterpie와 Bulbasaur의 공통 약점 속성 = {Fire, Flying}

```
$ python3 pokemon.py 1
{'id': 142, 'name': 'Aerodactyl', 'spawn_time': '23:40', 'type': ['Rock', 'Flying']}
{'id': 149, 'name': 'Dragonite', 'spawn_time': '23:38', 'type': ['Dragon', 'Flying']}
{'id': 126, 'name': 'Magmar', 'spawn_time': '20:36', 'type': ['Fire']}
```

Figure 4: command - python3 pokemon.py 1

B) 지우는 강력한 포켓몬을 잡고 싶어 한다. 진화를 많이 하면 강력하다고 판단한 지우는 포켓몬들을 최종진화체로 진화시키려 한다. 포켓몬이 진화하기 위해서는 사탕이 필요한데, 얼마나 사탕이 필요한지 계산해보려 한다.
각각의 포켓몬들은 진화 이전에 무슨 포켓몬이였는지와(prev_evolution), 진화 이후에 무슨 포켓몬인지(next_evolution) 그리고 어떤 사탕(candy)을 몇개 먹었을 때 진화하는지(candy_count)에 대한 정보를 가지고 있다. 이를 이용하여 Figure 5과 같이 각각의 최종진화체 포켓몬 마다 어떤 사탕이 최대 몇개 필요한지 출력한다.

```
$ python3 pokemon.py 2
Blastoise => Squirtle Candy: 125
Butterfree => Caterpie Candy: 62
Beedrill => Weedle Candy: 62
Venusaur => Bulbasaur Candy: 125
Pidgeot => Pidgey Candy: 62
                                    ...
Gyarados => Magikarp Candy: 400
Vaporeon => Eevee Candy: 25
Flareon => Eevee Candy: 25
Jolteon => Eevee Candy: 25
Omastar => Omanyte Candy: 50
Kabutops => Kabuto Candy: 50
Dragonite => Dratini Candy: 125
```

Figure 5: command - python3 pokemon.py 2 (문서의 일부분 출력)

- prev_evolution 필드의 경우 해당 포켓몬이 어떤 포켓몬으로부터 진화할 수 있는지 나타낸다. (진화 전의 전단계 포켓몬 포함)
- next_evolution 필드의 경우 해당 포켓몬이 진화할 수 있는 모든 포켓몬을 나타낸다. (진화 한 포켓몬의 진화체 포함)
- 여러 하위 포켓몬이 한가지 상위 포켓몬으로 진화하는 경우는 없다.
- prev_evolution 필드가 없을 경우 진화하지 않은 포켓몬이다.
- 최종진화체의 경우 next_evolution 필드와 candy_count 필드가 없다.

3 Email Search Engine

E-mail의 검색 엔진 또한 보낸/받은 메일을 찾아보자.
Figure 6과 7, 8은 특정 input에 대한 실행 결과를 나타낸다.
database는 ds2, collection 이름은 enron이며 emails.json은 import가 되어 있다고 가정한다.

- Text Search는 Email의 제목, 본문, 순으로 검색 중요 가정
- 기본 언어는 영어로 하고, 대소문자는 구별하지 않음
- 검색 결과는 최신 순과 정확도 순으로 정렬할 수 있음

입력 조건

- 검색 조건은 , (comma) 콤마로 구분하며 keyword와 : 입력 후 조건으로 검색
- keword끼리는 / 로 구분하고 keyword는 *not, from, to, sort*
- sort와 from은 검색조건이 한개로 고정

```
$ python3 engine.py "Social / not:Network / sort:date"
sender              subject          text                                  date
councilofthemaer    REMINDER: Franci REMINDER: FRANCISCO BARRIO TERRAZAS,  2001-06-01 14:03
councilofthemaer    Francisco Gros - FRANCISCO GROSPresident of the Natio   2001-06-01 13:20
rob.bradley@enro    Philanthropy Ide Ken:The attached one pager describes  2000-10-05 01:51
jngoodman@ncpa.o    i2Technologies/S Ken:Yesterday, we received a copy of  2000-09-12 01:55
```

Figure 6: python3 engine.py ”Social / not: Network / sort: date”

```
$ python3 engine.py "from:robyn@layfam.com"
sender              subject          text                                  date
robyn@layfam.com    Nicholas B-day   Hello Mamie and Papi, Invitations ar  2002-01-08 23:48
```

Figure 7: python3 engine.py ”from: robyn@layfam.com”

```
$ python3 engine.py "to:cindy.olson@enron.com, greg.whalley@enron.com / Please / not:Attached, previously"
sender              subject          text                                  date
dorothy.barnes@e    Revised Speech L Please call if you note any changes   1999-10-14 06:33
joannie.williams    Management Commi Please call if you have any question  2001-10-15 06:19
ryan.seleznov@en    Employee Concern I wanted to assure all of you that t  2001-10-19 12:55
ken.rice@enron.c    Urgently Need Yo I am compiling two "Top Ten Lists" t  2001-05-30 09:52
mary.clark@enron    All-employee mee Ken, Greg and Mark:Our next all-empl  2001-09-26 16:59
jeffrey.mcmahon@    RE: E-mail for a Fine with me.  Let's get this out A   2001-12-07 11:48
p..dupre@enron.c    Need a Bright Te Ken/Greg:I wish you both the best as  2001-12-03 10:24
```

Figure 8: python3 engine.py ”to: cindy.olson@enron.com, greg.whalley@enron.com / Please / not: Attached, previously”