Summer Internship 3rd Project

Relocating File program using Redis+MySQL+Flask+asyncio (Redis+MySQL+Flask 를 이용한 비동기 파일 재배치 프로그램)

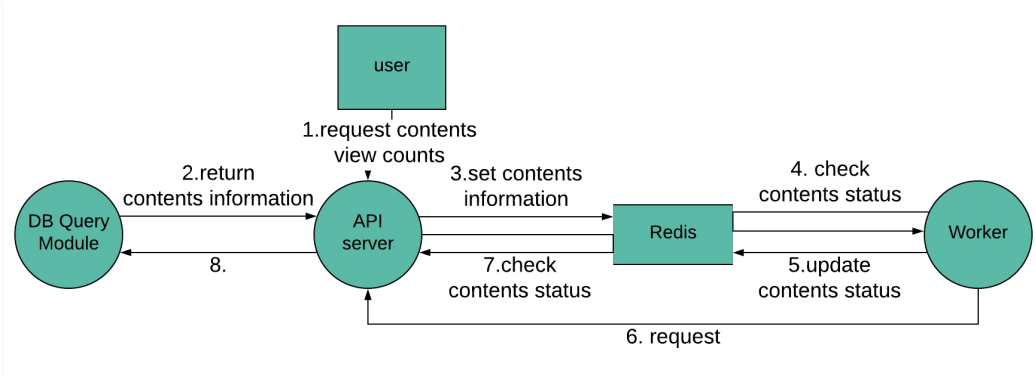
작성자 : 김지희

목차

---

1. 프로젝트 개요

- 프로젝트 기간
  2018.07.24 ~ 2018.08.02(10일)
- 개발 환경
  - OS : ubutu 16.04
  - python==3.6.5 , MySQL==5.7 , Flask==1.0.2 , PyMySQL==0.9.2 , redis-server==4.0.10
- 목적
  사용자에게 cid 와 counts를 받는 인터페이스를 제공하고 호출이 들어올 때마다 쿼리 모듈을 이용하여 MySQL database 의 정보를 가져오고 컨텐츠의 최종 위치를 정해 redis-server에 컨텐츠 정보를 json type으로 업데이트 한다. worker가 작업을 마치고 정해진 인터페이스로 API를 다시 호출하면 redis-server을 확인한 후 쿼리 모듈을 이용해 최종적으로 MySQL database를 업데이트한다.
- 기대효과
  사용자와 worker가 단 한번의 명령으로 MySQL, redis-server 의 정보를 받아오거나 업데이트하는 기능을 이용할 수 있는 인터페이스를 제공한다.
- Process



2. API 구현 정보

2.1 컨텐츠 정보 쿼리 및 redis 업데이트(Process step1 ~ step3)

2.1.1 API Specification

- **URL** host:port/post_sentence
- **Method:**

  `POST`
- **Data Params**

  `{cid=7&count=664}`
- **Success Response:**
  - **Code:** `200`
  - **Content:**

    `{"cid": "7", "count": "1964", "target": "silver", "db_level": "bronze", "filename": "g.mp4", "worker_id": null, "status": "update"}`
- **Faults Response:**
  - **Code:** `404`
  - **Content:**
    - Not found
    - Non existent URI
  - **Code:** `500`
  - **Content:**
    - Internal Server error (MySQL, Redis-server error etc.)

- **Sample Call:**

```
foo@bar:~/$ curl http://192.168.10.108:5000/post_sentence -d "cid=7&count=1964"
{"cid": "7", "count": "1964", "target": "silver", "db_level": "bronze",
"filename": "g.mp4", "worker_id": null, "status": "update"}
```

2.1.2 기능 설명

1. 사용자가 컨텐츠를 조회하면 컨텐츠의 cid 와 count 정보를 포함하여 API를 호출 (e.g.curl http://192.168.10.108:5001/post_sentence -d "cid=3&count=664")
2. db_query 모듈을 이용하여 database의 contents table 과 level table에서 post 된 cid를 가진 content의 현재 위치와 목적 위치를 반환
3. redis-server에 연결하여 post 요청이 들어온 cid에 대한 count, 목적 위치, 현재 위치, filename, worker_id, status 정보를 json type으로 만들어 key:cid, value :json 형식의 정보 로 redis database에 set 한 후 value를 반환 (e.g.) `{'3':{"cid": "3", "count": "664", "target": "bronze", "db_level": "silver", "filename": "c.mp4","worker_id": null, "status": "update"}}` 형식으로 저장

worker_id는 worker에서 지정되므로 null로 초기화하며 status는 현재 위치와 목적 위치가 같은 경우 'done', 다른 경우 'update'로 초기화

contents table

#	cid	content_level	filename	generate_time	update_time
1	1	gold	a.mp4	2018-07-29 …	2018-08-02 18:30:17
2	2	bronze	b.mp4	2018-07-29 …	2018-08-02 10:47:40
3	3	silver	c.mp4	2018-07-29 …	2018-08-02 18:30:18
4	4	bronze	d.mp4	2018-07-29 …	2018-08-01 16:25:19
5	5	silver	e.mp4	2018-07-29 …	2018-08-02 18:30:18
6	6	gold	f.mp4	2018-07-29 …	2018-08-01 17:55:23

level table

#	content_level	max_counts	path	min_counts
1	bronze	999	/etc/inisoft/redis_project/bronze/	0
2	gold	3001	/home/inisoft/workspace/inisoft/redi…	2000
3	silver	1999	/var/www/html/redis_project/silver/	1000

2.2 redis 값 체크 및 MySQL database 업데이트(Process step6 ~ step8)

2.2.1 API Specification

- **URL** host:port/update_sentence
- **Method:**

`POST`

- **Data Params**

  `{cid:7}`

- **Success Response:**

  - **Code:** `200`
  - **Content:** `7`

- **Faults Response:**

  - **Code:** `404`
  - **Content:**
    - Not found
    - Non existent URI
  - **Code:** `500`
  - **Content:**
    - Internal Server error (MySQL, Redis-server error etc.)

- **Sample Call:**

```
foo@bar:~/$ curl http://192.168.10.108:5000/update_sentence -d "cid=7"
7
foo@bar:~/$ curl http://192.168.10.108:5000/update_sentence -d "cid=8"
check your status again
```

2.2.2 기능 설명

1. worker가 파일을 재배치한 후 해당 contents 의 status 를 'done' 으로 바꾸고 API 를 호출 (e.g. curl http://192.168.10.108:5000/update_sentence -d "cid=3")
2. request를 받으면 cid 를 Key 값으로 redis에서 해당 content의 status 가 'done' 인지 검사하고 MySQL의 contents table 에 새로운 level 과 update time 을 업데이트 만약 status 가 'done' 이 아니면 "check your status again" 메세지를 반환

   redis status check 후 content_level과 update_time update 결과

#	cid	content_level	filename	generate_time	update_time
7	7	silver	g.mp4	2018-07-29 ...	2018-08-06 17:30:16

---

3. Issue 발생 및 해결방법 기록

1. 디렉터리 쓰기 권한 설정
    - issue : 파일이 이동해야 하는 route가 사용자에게 쓰기 권한이 없을 경우 파일 이동 시 'permission denied' 에러 발생
    - 해결 방안 :

```
foo@bar$ sudo chown foo:foo -R 업로드 되는 폴더 path
```

2. remote database 접근 권한
    - issue : remote server 에 있는 mysql을 local에서 PyMySQL 으로 연결 불가하여 sshtunneling을 이용하여 보안 및 네트워크 비용 증가
    - 해결 방안 :remote server 의 MySQL access 권한 및 config을 수정하여 sshtunneling 삭제

```
foo@bar$ mysql -u -p
mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'password';
```

3. redis input & output 형식
    - issue : redis 에 json 형식으로 input 하면 byte string으로 저장되어 get 할 때 decode + json type 변경 필요 , None 값이 존재하여 jsonify 이용 불가
    - 해결 방안:

```
rc.set(data['cid'],json.dumps(data).encode('utf-8'))
```

4. min_count, max_count를 통한 get_content_level 함수 scalability 확장
    - issue : 기존 코드 : count값을 받아 file 의 최종 위치 반환 시 content_level 이 gold, silver, bronze 3개 일 때만 정상적으로 실행됨(Database의 level table에 counts 칼럼만 존재)

```
def get_target(count):
db = db_query.db()
a = db.select(table='level', column ='*', order_by = 'counts desc' )
if count >= a[0]['counts']:
    level = a[0]['content_level']
elif count >= a[1]['counts']:
    level = a[1]['content_level']
else:
    level = 'bronze'
return level
```

    - 해결 방안 : Database의 level table에 max_counts 와 min_counts 칼럼을 추가하여 Database만 수정하면 level 추가가 가능하고, database 에 저장된 level중 max_counts 가 가장 큰 level의 max_count 보다 더 큰 값이 들어올 경우 자동으로 database의 max_count 값을 수정하도록 변경

```
def get_target(count):
db = db_query.db()
db.check_max_count(count)
level_table = db.select(table='level', column ='*' )
for i in level_table:
    if int(i['max_counts'])>= count>= int(i['min_counts']) :
        level = i['content_level']
        return level
```

---

4. 프로젝트 후기

개선할점

- scalability와 유지보수 를 고려하여 하드코딩, 반복되는 코드를 최소화 해야한다.
- 더 많은 에러 발생 경우에 대한 처리 방법이 추가되어야 한다.

느낀점

내가 구현한 API의 기능이 까다롭거나 개발 난이도가 높지는 않았지만 과제를 받을 당시에는 API 개념이나 개발 목적에 대한 지식이 부족하여 어떤 기능이 있어야 하는지조차도 모호했다. 이 때문에 개발 자체보다 개발을 위해 공부하고 소통하는 과정에서 더 많이 고민했던 것 같다. 공부하고 팀원들과 회의를 하는 과정에서 요구 사항이 명확해져서 결국 예상보다 빠르게 프로젝트를 끝낼 수 있었다. 개발 중 겪은 어려움으로는 remote server에서 이용되는 library나 module의 가용성을 계속 체크해야 해서 시간이 소요되었고 권한 문제나 보안 상의 이유로 파일의 설정 또한 로컬에서 단독으로 개발 할 때보다 복잡했다. 이 프로젝트를 통해 API 와 같이 이전에 접해보지 못한 개념에 대해 제대로 공부할 수 있어서 좋았고 에러 해결을 하면서 구글링 실력이 향상된 것을 느낄 수 있었다.