

## Programming Assignment #2

Due : 27th Oct. (Sun), 11:59 PM

### 1. Introduction

FILE I/O 시스템콜을 사용하여 메모리에 있는 데이터를 파일로 저장해본다.

### 2. Problem specification

Key-Value(KV) 데이터베이스는 간단한 Key-Value 함수를 사용하여 데이터를 저장하는 비관계형 데이터베이스이다.

이전 과제에서는 Key와 Value를 모두 메모리에 있는 Hash Table 자료구조에 저장했다. 데이터들을 모두 메모리에 가지고 있기에는 메모리의 용량이 부족할 뿐만 아니라 시스템이 갑자기 종료되는 경우 데이터들을 모두 잃게 되는 문제가 있다.

이번 과제에서는 메모리에 가지고있는 KV쌍이 일정 수를 넘어가면 그 데이터들을 파일로 적고 메모리를 해제하도록 한다. 이전 과제에서 추가되는 기능들은 다음과 같다.

**db\_t\* db\_open(int size);**

- 현재 디렉토리 위치에 파일을 저장할 "./db" 디렉토리를 생성한다.
- (이미 "./db" 가 생성되어 있는 경우 해당 디렉토리를 사용한다)

**void db\_close(db\_t\* db);**

- 데이터베이스에서 사용했던 자원들을 모두 해제하고, Hash Table에 데이터가 남아있을 경우 "./db" 디렉토리 밑의 하나의 파일로 저장한다

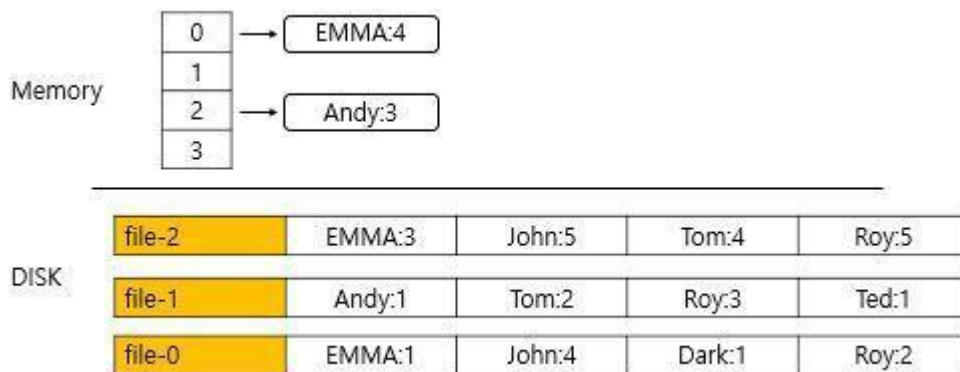
**void db\_put(db\_t\*db, char\* key, int keylen, char\* val, int vallen);**

- 현재 Hash Table 위의 KV쌍의 수가 size와 같으며 메모리 상에 없는 KV쌍이 추가될 때, 기존 Hash Table위의 데이터들을 모두 "./db" 디렉토리 밑에 한 개의 파일로 저장한다.
- 파일로 저장한 Hash Table의 데이터들은 모두 free하고 새로운 KV쌍을 추가한다.

**char\* db\_get(db\_t\* db, char\* key, int keylen, int\* vallen);**

- 주어진 key가 메모리에 존재하지 않다면, 데이터들을 저장했던 파일들 중에서 해당 key에 맞는 value를 찾아 반환한다.
- 주어진 key가 존재하지 않는다면 NULL을 반환한다.

다음은 Hash Table의 크기를 4로 설정했을 때 전체적인 데이터베이스의 구조를 나타내는 예시이다.



Hash Table에 KV의 쌍이 4개 존재하는 경우를 가정해보자.

- 현재 Hash Table에 존재하는 Key에 대한 **db\_put**
  - ✓ 메모리 상에 있는 해당 Key에 대한 Value값을 업데이트 한다.
- 현재 Hash Table위에 없는 Key에 대한 **db\_put**
  - ✓ 기존 Hash Table위에 있던 4개의 KV쌍의 정보를 한 파일에 저장한 후, 메모리 상에서 free시킨다. Hash Table 새로운 Key에 대한 KV쌍을 만든다.

위 그림은 file-0, file-1, file-2순으로 파일이 저장되고 있는 그림이다. 파일 간의 중복되는 데이터들이 존재하지만, 최신 데이터는 메모리, file-2, file-1, file-0 순으로 존재함을 확인할 수 있다.

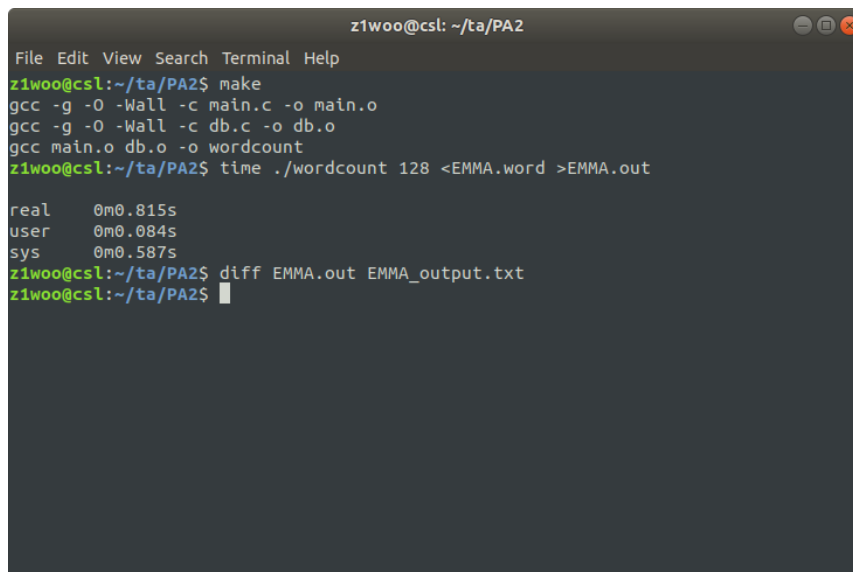
파일 이름과 KV쌍을 저장하는 파일의 구조는 자율이나, 여러 파일들 중 최신 데이터들을 찾아내는 방법과, 직접 구현한 파일 구조를 보고서에 상세히 기술해야 한다.

### 3. Example

<pre> \$./wordcount 128 EMMA EMMA JOHN </pre>	<pre> DB opened GET [EMMA] [NULL] PUT [EMMA] [1] GET [EMMA] [1] PUT [EMMA] [2] GET [JOHN] [NULL] PUT [JOHN] [1] DB closed </pre>
<pre> \$./wordcount 128 EMMA </pre>	<pre> DB opened GET [EMMA] [2] PUT [EMMA] [3] DB closed </pre>

## 4. Hand in instructions

- 본 과제는 개인 과제이며 icampus의 과제 란에 제출한다.
- 과제 제출 시간은 icampus 제출 시각 기준으로 하며, 매 24시간마다 25%씩 감점된다.
- 제출시 프로그램코드와 보고서를 함께 압축하여 제출한다.
  - ✓ 압축 파일은 Makefile, main.c, db.c, db.h, README.pdf 로 이루어져 있어야 하며, 압축 파일의 이름과 확장자는 "학번.tar.gz"여야 한다.
  - ✓ 프로그램 코드와 별도로, 구현에 대한 내용을 담은 보고서를 함께 제출한다. 보고서의 파일 포맷은 pdf로 제한하며, 그 외의 형식에는 제한이 없다. 제목은 README.pdf 로 한다.
- 본 과제는 혼자서 한다.
- 본 과제에는 성능점수가 포함된다.
  - ✓ 상위 30%내에 들 경우 10점, 상위 30%이상은 5점
  - ✓ 워크스테이션실의 PC에서 Virtual Box 기준 10분 이상이 걸릴 경우 성능 점수를 받을 수 없다.
  - ✓ 성능측정은 다음과 같이 진행된다



```
z1woo@csl: ~/ta/PA2
File Edit View Search Terminal Help
z1woo@csl:~/ta/PA2$ make
gcc -g -O -Wall -c main.c -o main.o
gcc -g -O -Wall -c db.c -o db.o
gcc main.o db.o -o wordcount
z1woo@csl:~/ta/PA2$ time ./wordcount 128 <EMMA.word >EMMA.out

real    0m0.815s
user    0m0.084s
sys     0m0.587s
z1woo@csl:~/ta/PA2$ diff EMMA.out EMMA_output.txt
z1woo@csl:~/ta/PA2$
```

- Copy 할 경우, 연구실 자체 규정에 따라 처벌하며, 상당한 불이익이 있을 수 있다.
  - ✓ 과제 Copy 또는 미제출 2회이상 F

## 5. Additional

GNU make 도구는 큰 프로그램을 만들 때 유용하게 사용된다. 이는 프로그램을 제작하기 위해 어떤 코드를 (재)컴파일해야 하는지 결정해준다. Makefile이 준비되면, 어떤 소스 코드를

변경하던지 셸에서 단순히 make란 명령을 실행시키는 것으로 재 컴파일이 필요한 모든 파일을 알아서 찾아 다시 컴파일한다.