

Implement a simple database using memory mapped files

Igor Półchłopek



The project description

- A program that make a simple database using memory mapped files.
- Program can print, add, delete and replace record from tables(files) in database.
- Everything is controlled by console interface.

The challenge

- Support memory mapped files as I/O for my simple database is little tricky at beginning.
- Deleting records wasn't that obvious and I wasted time on trying make delete process too complicated.

The solution

```
201 void menu(){
202     char c = '0';
203     int i;
204     printf("\nEnter nubmer:\n1.Print table 'person'\n2.Print table 'address'\n3.Add to table 'person'\n4.Add to table 'address'\n5.Delete from table 'person'\n6.Delete from table 'address'\n7.Exit\n");
205     if (scanf("%d", &i) == 0 || i < 0 || i > 7) {
206         printf("Invalid input\n");
207     }
208     switch(i){
209         case(1):
210             printPerson();
211             menu();
212             break;
213         case(2):
214             printAddress();
215             menu();
216             break;
217         case(3):
218             puts(addPerson());
219             menu();
220             break;
221         case(4):
222             puts(addAddress());
223             menu();
224             break;
225         case(5):
226             deletePerson();
227             menu();
228             break;
229         case(6):
230             deleteAddress();
231             menu();
232             break;
233         case(7):
234             exit(0);
235     }
236 }
```

Menu was made by simple switch case and scanf.



```
243     case(8):
244         puts(replaceAddress());
245         menu();
246         break;
247     case(9):
248         printPersonAddress();
249         menu();
250         break;
251     case(0):
252         break;
253     default:
254         break;
255 }
256 }
```

The solution

```
void printPerson(){
    puts("\nTable person:\n[id]      [firstName]      [lastName]      [age]      [gender]");
    const int fd = open("mmfiles/person.db", O_RDONLY, S_IRUSR | S_IWUSR );
    struct stat fileInfo;
    if (fstat(fd, &fileInfo) == -1){
        perror("Error getting the file size");
        exit(EXIT_FAILURE);
    }

    if (fileInfo.st_size == 0){
        fprintf(stderr, "Error: File is empty, nothing to do\n");
        exit(EXIT_FAILURE);
    }
    const int n_records = fileInfo.st_size/sizeof(struct Person);

    struct Person* person = mmap(0, fileInfo.st_size, PROT_READ, MAP_PRIVATE, fd, 0);
    if ( person == MAP_FAILED ) {
        perror("mmap failed");
        exit(-1);
    }
    for ( int r = 0; r < n_records; ++r )
        if((person+r)->ID != 0)
            printf( "%d\t%-19s%-18s%-14d%c\n",
                (person+r)->ID, (person+r)->firstName, (person+r)->lastName, (person+r)->age, (person+r)->gender );

    if ( munmap( person, fileInfo.st_size) ) {
        perror("unmap");
        exit(-1);
    }
    close(fd);
}
```

Printing – for either files
either functions

The solution

```
365     personID += 1;
366     const int fd = open("mmfiles/person.db", O_RDWR, S_IRUSR | S_IWUSR );
367     struct stat fileInfo;
368     if (fstat(fd, &fileInfo) == -1){
369         perror("Error getting the file size");
370         exit(EXIT_FAILURE);
371     }
372
373     off_t DBsize = sizeof(struct Person) + fileInfo.st_size;
374     int n_records = fileInfo.st_size/sizeof(struct Person);
375     const int status = ftruncate( fd, DBsize );
376     if ( status != 0 ) {
377         perror("ftruncate");
378         exit(-1);
379     }
380     struct Person* person = mmap(0, DBsize, PROT_WRITE, MAP_SHARED, fd, 0);
381     if ( person == MAP_FAILED ) {
382         perror("mmap failed");
383         exit(-1);
384     }
385
386     fillPerson(person+n_records, firstName, lastName, age, gender);
387
388     if ( munmap( person, DBsize ) ) {
389         perror("unmap");
390         exit(-1);
391     }
392     close(fd);
393
394     return "##### Person Added #####";
395 }
```

```
329 // ##### Enter data #####
330 printf("Add record to table person:\n");
331 int age;
332
333 puts("Enter age(0-122):");
334 scanf("%d", &age);
335 if (age<0 || age>122) { //122, because oldest man on earth was 122 yo :D.
336     puts("You entered wrong age.");
337     return "Wrong values";
338 }
339
340 char gender;
341 puts("Enter gender(F/M): ");
342 scanf(" %c", &gender);
343 if (gender != 'M' && gender != 'F') { // M - male, F - female
344     puts("You entered wrong gender.");
345     return "Wrong values";
346 }
347
348 char firstName[30];
349 puts("Enter first name(letters in length: 2 - 30):");
350 scanf("%s", firstName);
351 if (strlen(firstName) < 2 || strlen(firstName) > 30 || checkString(firstName)) {
352     puts("You entered wrong firstName.");
353     return "Wrong values";
354 }
355
356 char lastName[30];
357 puts("Enter last name(letters in length: 2 - 30):");
358 scanf("%s", lastName);
359 if (strlen(lastName) < 2 || strlen(lastName) > 30 || checkString(lastName)) {
360     puts("You entered wrong lastName.");
361     return "Wrong values";
362 }
363
364 // #####
```

Adding – for either files
either functions

Info from user
console input

The solution

```
473 void deletePerson(){
474     puts("Enter ID of record u want to delete:");
475     int delID;
476     scanf("%d", &delID);
477     const int fd = open("mmfiles/person.db", O_RDWR, S_IRUSR | S_IWUSR );
478     struct stat fileInfo;
479     if (fstat(fd, &fileInfo) == -1){
480         perror("Error getting the file size");
481         exit(EXIT_FAILURE);
482     }
483
484     struct Person* person = mmap(0, fileInfo.st_size, PROT_WRITE, MAP_SHARED, fd, 0);
485     if ( person == MAP_FAILED ) {
486         perror("mmap failed");
487         exit(-1);
488     }
489
490     bzero(person+delID-1, sizeof(struct Person));
491
492     if ( munmap( person, fileInfo.st_size ) ) {
493         perror("unmap");
494         exit(-1);
495     }
496     close(fd);
497     puts("\n##### Person Deleted #####\n");
498 }
```

Deleting – for either files
either functions

The solution

```
567 const int fd = open("mmfiles/person.db", O_RDWR, S_IRUSR | S_IWUSR );
568 struct stat fileInfo;
569 if (fstat(fd, &fileInfo) == -1){
570     perror("Error getting the file size");
571     exit(EXIT_FAILURE);
572 }
573
574 struct Person* person = mmap(0, fileInfo.st_size, PROT_WRITE, MAP_SHARED,
575 if ( person == MAP_FAILED ) {
576     perror("mmap failed");
577     exit(-1);
578 }
579
580 const int n_records = fileInfo.st_size/sizeof(struct Person);
581
582 fillPerson(person+repID-1, firstName, lastName, age, gender);
583 (person+repID-1)->ID = repID;
584
585 if ( munmap( person, fileInfo.st_size ) ) {
586     perror("unmap");
587     exit(-1);
588 }
589 close(fd);
590
591 return "##### Person Replaced #####";
592 }
```

```
329 // ##### Enter data #####
330 printf("Add record to table person:\n");
331 int age;
332
333 puts("Enter age(0-122):");
334 scanf("%d", &age);
335 if (age<0 || age>122) { //122, because oldest man on earth was 122 yo :D.
336     puts("You entered wrong age.");
337     return "Wrong values";
338 }
339
340 char gender;
341 puts("Enter gender(F/M): ");
342 scanf(" %c", &gender);
343 if (gender != 'M' && gender != 'F') { // M - male, F - female
344     puts("You entered wrong gender.");
345     return "Wrong values";
346 }
347
348 char firstName[30];
349 puts("Enter first name(letters in length: 2 - 30):");
350 scanf("%s", firstName);
351 if (strlen(firstName) < 2 || strlen(firstName) > 30 || checkString(firstName)) {
352     puts("You entered wrong firstName.");
353     return "Wrong values";
354 }
355
356 char lastName[30];
357 puts("Enter last name(letters in length: 2 - 30):");
358 scanf("%s", lastName);
359 if (strlen(lastName) < 2 || strlen(lastName) > 30 || checkString(lastName)) {
360     puts("You entered wrong lastName.");
361     return "Wrong values";
362 }
363
364 // #####
```

Replacing – for either files
either functions

The solution

```
667 void printPersonAddress(){
668     puts("Enter ID of person whose address you want to print:");
669     int chosenPersonID;
670     scanf("%d", &chosenPersonID);
671     puts("\nTable address:\n[id]      [personId]      [city]      [postalCode]      [street]      [nr]");
672     const int fd = open("mmfiles/address.db", O_RDONLY, S_IRUSR | S_IWUSR );
673     struct stat fileInfo;
674     if (fstat(fd, &fileInfo) == -1){
675         perror("Error getting the file size");
676         exit(EXIT_FAILURE);
677     }
678     if (fileInfo.st_size == 0){
679         fprintf(stderr, "Error: File is empty, nothing to do\n");
680         exit(EXIT_FAILURE);
681     }
682     const int n_records = fileInfo.st_size/sizeof(struct Address);
683     struct Address* address = mmap(0, fileInfo.st_size, PROT_READ, MAP_PRIVATE, fd, 0);
684     if ( address == MAP_FAILED ) {
685         perror("mmap failed");
686         exit(-1);
687     }
688     for ( int r = 0; r < n_records; ++r )
689         if((address+r)->personID == chosenPersonID)
690             printf( "%d\t%-14d%-14s%-16s%-16s\n",
691                 (address+r)->ID, (address+r)->personID, (address+r)->city, (address+r)->postalCode, (address+r)->street, (address+r)->nr );
692     if ( munmap( address, fileInfo.st_size ) ) {
693         perror("unmap");
694         exit(-1);
695     }
696 }
697 close(fd);
698 }
```

Primitive linking(just printing
right records)

Result

```
igor@igor-Aspire-R5-571T:~/Pulpit/STUDIA/UNIX$ ./out.sh
```

Enter nubmer:

- 1.Print table 'person'
- 2.Print table 'address'
- 3.Add to table 'person'
- 4.Add to table 'address'
- 5.Delete from table 'person'
- 6.Delete from table 'address'
- 7.Replace in table 'person'.
- 8.Replace in table 'address'
- 9.Print address for personID
- 0.Exit

Your choice: 1

Table person:

[id]	[firstName]	[lastName]	[age]	[gender]
1	John	Lenon	34	M
4	Filip	Pol	26	M
5	Jacek	Kowalski	44	M
6	Igor	Pol	66	M
7	Karol	Nowak	45	F
8	Tomek	Gromek	45	M

Enter nubmer:

- 1.Print table 'person'
- 2.Print table 'address'
- 3.Add to table 'person'
- 4.Add to table 'address'
- 5.Delete from table 'person'
- 6.Delete from table 'address'
- 7.Replace in table 'person'.
- 8.Replace in table 'address'
- 9.Print address for personID
- 0.Exit

Your choice: 3

Add record to table person:

Enter age(0-122):

█

Hardest issue

- Handling memory mapped files was necessary for this project.
- A lot time I spend on interface and user input verification
- Hardest was implementing „delete”. It wasn't obvious to do that way as I did.