

# Concepts Avancés de Bases de données

Joaquim LEFRANC et Jérôme Skoda

October 15, 2017

## Abstract

Comment avons nous pu écrire autant de chose sur seulement deux boucles imbriqués et une fonction de merge de moins de 20 lignes?

## 1 Comment compiler le projet

### 1.1 Avec le terminal

- make all : Compile tout les fichiers
- make test : Lancement de la série de tests automatiques
- make doc : Génération de la documentation (doxygen)
- make rapport : Génération du rapport (latex)
- make clean : Nettoyage du projet (supression des objets et binaires)
- make demo-tp1 : Lancer la démo tp1
- make demo-tp2 : Lancer la démo tp2
- make rm-rs : Supprime le fichier res/RS.txt

### 1.2 Avec ECLIPSE

Pour lancer une commande utiliser les boutons magique build targets.

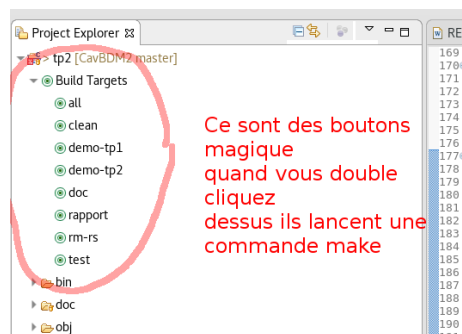


Figure 1: Double clique ici!

### 1.3 Avec code:block

Aucun portage sur code:block de prévu. Nous avons passé une longue heure à configurer eclipse et nous en avons assez!

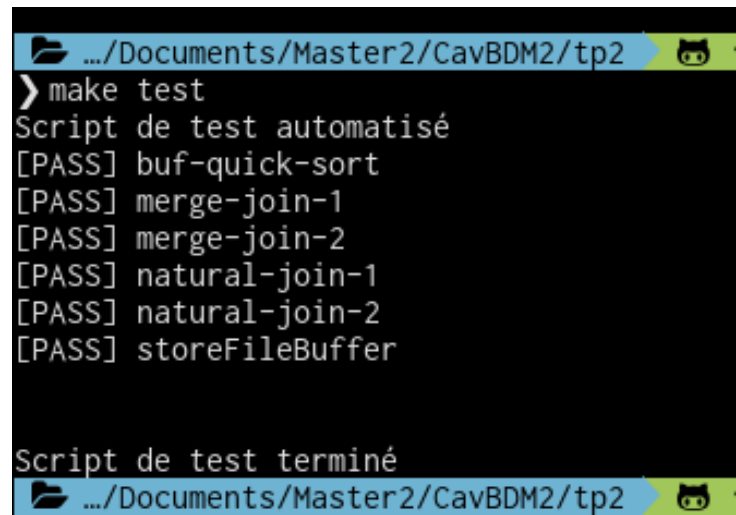
## 2 Arborescence

- bin : Binaire exécutable
  - demo : Exécutable de démonstration
  - test : Exécutable de test
- doc : Documentation doxygen sous differents formats
- rapport : Source du rapport
- res : Ressources necessaire au projet (fichier de bdd)
- script : Script utilisé pour les test
- src : Source du projet
  - bdd : Source de la bibliothèque
  - demo : Sources des differentes démonstrations d'utilisation
  - test : Sources des dufferents tests
- sujet.pdf : Sujet du projet
- README.md : Le readme du projet
- rappot.pdf : C'est moi

## 3 Caracteristiques

- Le code est organisé
- Il y a des code des tests
- Il y a la doc
- Il y a un rapport
- Et il y a pleins d'autre chose

## 4 Test unitaire

A terminal window with a black background and white text. The title bar at the top shows the file path ".../Documents/Master2/CavBDM2/tp2" and a GitHub icon. The terminal content shows a command prompt followed by "make test", which triggers an automated test script. The script lists six tests, all of which pass, indicated by "[PASS]" prefixes. The tests are: buf-quick-sort, merge-join-1, merge-join-2, natural-join-1, natural-join-2, and storeFileBuffer. After the tests, the message "Script de test terminé" is displayed. The terminal window ends with the same title bar as the beginning.

```
> make test
Script de test automatisé
[PASS] buf-quick-sort
[PASS] merge-join-1
[PASS] merge-join-2
[PASS] natural-join-1
[PASS] natural-join-2
[PASS] storeFileBuffer

Script de test terminé
```

## 1 Navigation

- **bdd** : Source de la BDD
- **demo** : Exemple d'utilisation de la BDD
- **test** : Test unitaire de la BDD

## 2 Index des classes

### 2.1 Liste des classes

Liste des classes, structures, unions et interfaces avec une brève description :

<b>buf</b>	
Structure de donnée représentant un buffer	<b>1</b>

## 3 Index des fichiers

### 3.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

<b>src/bdd/bdd.c</b>	<b>3</b>
<b>src/bdd/bdd.h</b>	<b>3</b>
<b>src/bdd/hexdump.c</b>	<b>8</b>
<b>src/bdd/hexdump.h</b>	<b>8</b>
<b>src/bdd/quicksort.c</b>	<b>9</b>
<b>src/bdd/quicksort.h</b>	<b>10</b>
<b>src/demo/1-natural-join.c</b>	<b>10</b>
<b>src/demo/2-merge-join.c</b>	<b>11</b>
<b>src/test/storeFileBuffer.c</b>	<b>11</b>

## 4 Documentation des classes

### 4.1 Référence de la structure buf

Structure de donnée représentant un buffer.

**Attributs publics**

- char \* **v**  
*Value: Valeur.*
- size\_t **s**  
*Size: Taille.*
- size\_t **c**  
*Count: Nombre de valeur entrée.*

**4.1.1 Description détaillée**

Structure de donnée représentant un buffer.

TP n°: 2

Titre du TP : Merge Join

Date : 13/10/17

Nom : Lefranc Prenom : Joaquim email : [lefrancjoaquim@gmail.com](mailto:lefrancjoaquim@gmail.com)

Nom : Skoda Prenom : Jérôme email : [contact@jeromeskoda.fr](mailto:contact@jeromeskoda.fr)

Remarques :

History: 8871b16 [PASS] 5b1768f [PASS]

**4.1.2 Documentation des données membres****4.1.2.1 c**

size\_t buf::c

Count: Nombre de valeur entrée.

**4.1.2.2 s**

size\_t buf::s

Size: Taille.

## 4.1.2.3 v

```
char* buf::v
```

Value: Valeur.

La documentation de cette structure a été générée à partir du fichier suivant :

- [src/bdd/bdd.c](#)

## 5 Documentation des fichiers

### 5.1 Référence du fichier src/bdd/bdd.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "bdd.h"
#include "hexdump.h"
```

Graphe des dépendances par inclusion de bdd.c:

### 5.2 Référence du fichier src/bdd/bdd.h

```
#include <stdio.h>
#include "quicksort.h"
```

Graphe des dépendances par inclusion de bdd.h: Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

#### Fonctions

- struct [buf](#) \* [buf\\_create](#) (size\_t size)
- void [buf\\_destroy](#) (struct [buf](#) \*buf)
- char [buf\\_put](#) (struct [buf](#) \*buf, char value)
- void [storeFileBuffer](#) (FILE \*fp, struct [buf](#) \*buf)
 

*Enregistre un le 1er caractère de chaque ligne d'un fichier dans un buffer.*
- void [writeBufferInFile](#) (FILE \*fp, const struct [buf](#) \*buf)
- void [natural\\_join](#) (const struct [buf](#) \*buf\_a, const struct [buf](#) \*buf\_b, struct [buf](#) \*buf\_out)
- void [merge\\_join](#) (const struct [buf](#) \*buf\_a, const struct [buf](#) \*buf\_b, struct [buf](#) \*buf\_out)
 

*Merge join.*
- void [buf\\_quicksort](#) (struct [buf](#) \*buf)
 

*Quick sort buffer.*
- void [buf\\_dump](#) (const struct [buf](#) \*buf)
 

*Hex dump buffer.*
- struct [buf](#) \* [storeFileBufferOC](#) (const char \*file\_name, size\_t buffer\_size)
 

*Ouvre un fichier Stoque son contenu dans un buffer Ferme le fichier.*
- char [writeBufferInFileOC](#) (const char \*file\_name, const struct [buf](#) \*)
 

*Ouvre un fichier Ecrit le contenu du buffer dans le fichier Ferme le fichier.*

### 5.2.1 Documentation des fonctions

#### 5.2.1.1 buf\_create()

```
struct buf* buf_create (
    size_t size )
```

Creation d'un buffer

**Paramètres**

in	size	Taille du buffer
----	------	------------------

**5.2.1.2 buf\_destroy()**

```
void buf_destroy (
    struct buf * buf )
```

Détruit le buffer

**Paramètres**

in	buf	buffer à détruire
----	-----	-------------------

**5.2.1.3 buf\_dump()**

```
void buf_dump (
    const struct buf * buf )
```

Hex dump buffer.

**Paramètres**

in	buf	buffer à trier
----	-----	----------------

**5.2.1.4 buf\_put()**

```
char buf_put (
    struct buf * buf,
    char value )
```

Ajoute un caractere dans le buffer s'il reste de la place

**Paramètres**

in	buf	buffer d'entrée
in	value	valeur à entrer

**Renvoie**

0 succès -1 erreur: buffer plein



### 5.2.1.5 buf\_quicksort()

```
void buf_quicksort (
    struct buf * buf )
```

Quick sort buffer.

#### Paramètres

	<i>buf</i>	Trie le buffer en entrée
in	<i>buf</i>	buffer à trier

### 5.2.1.6 merge\_join()

```
void merge_join (
    const struct buf * buf_a,
    const struct buf * buf_b,
    struct buf * buf_out )
```

Merge join.

#### Paramètres

in	<i>buf_a</i>	relation externe
in	<i>buf_b</i>	relation interne
out	<i>buf_out</i>	resultat du merge_join
in	<i>buf_a</i>	relation a
in	<i>buf_b</i>	relation b
out	<i>buf_out</i>	resultat du merge_join

### 5.2.1.7 natural\_join()

```
void natural_join (
    const struct buf * buf_a,
    const struct buf * buf_b,
    struct buf * buf_out )
```

Natural join

#### Paramètres

in	<i>buf_a</i>	relation externe
in	<i>buf_b</i>	relation interne
out	<i>buf_out</i>	resultat du natural join de buf_a et buf_b

### 5.2.1.8 storeFileBuffer()

```
void storeFileBuffer (
    FILE * fp,
    struct buf * buf )
```

Enregistre un le 1er caractère de chaque ligne d'un fichier dans un buffer.

#### Paramètres

in	<i>Fichier</i>	d'entrée
out	<i>Buffer</i>	de sortie

### 5.2.1.9 storeFileBufferOC()

```
struct buf* storeFileBufferOC (
    const char * file_name,
    size_t buffer_size )
```

Ouvre un fichier Stoque son contenu dans un buffer Ferme le fichier.

#### Paramètres

in	<i>File_name</i>	Fichier
in	<i>buffer_size</i>	Taille du buffer, si trop petit pour contenir le fichier entièrement alors le reste du fichier est ignoré

#### Renvoie

buffer si NULL alors erreur de lecture du fichier

### 5.2.1.10 writeBufferInFile()

```
void writeBufferInFile (
    FILE * fp,
    const struct buf * buf )
```

Ecrit un buffer dans un fichier

#### Paramètres

out	<i>fp</i>	fichier de sortie
in	<i>buf</i>	fichier d'entrée

#### 5.2.1.11 writeBufferInFileOC()

```
char writeBufferInFileOC (
    const char * file_name,
    const struct buf * buf )
```

Ouvre un fichier Ecrit le contenu du buffer dans le fichier Ferme le fichier.

##### Paramètres

in	<i>File_name</i>	Fichier
in	<i>buf</i>	Buffer à écrire dans le fichier

##### Renvoie

-1 si erreur dans l'ouverture du fichier

### 5.3 Référence du fichier src/bdd/hexdump.c

```
#include <stdio.h>
```

Graphe des dépendances par inclusion de hexdump.c:

##### Fonctions

- void [hexDump](#) (char \*desc, const void \*addr, int len)

#### 5.3.1 Documentation des fonctions

##### 5.3.1.1 hexDump()

```
void hexDump (
    char * desc,
    const void * addr,
    int len )
```

### 5.4 Référence du fichier src/bdd/hexdump.h

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

##### Fonctions

- void [hexDump](#) (char \*desc, const void \*addr, int len)

### 5.4.1 Documentation des fonctions

#### 5.4.1.1 hexDump()

```
void hexDump (
    char * desc,
    const void * addr,
    int len )
```

### 5.5 Référence du fichier src/bdd/quicksort.c

```
#include <stdio.h>
#include <stdlib.h>
```

Graphe des dépendances par inclusion de quicksort.c:

#### Fonctions

- void [swap](#) (char \*x, char \*y)
- int [choose\\_pivot](#) (char i, char j)
- void [quicksort](#) (char \*list, int m, int n)

### 5.5.1 Documentation des fonctions

#### 5.5.1.1 choose\_pivot()

```
int choose_pivot (
    char i,
    char j )
```

#### 5.5.1.2 quicksort()

```
void quicksort (
    char * list,
    int m,
    int n )
```

### 5.5.1.3 swap()

```
void swap (
    char * x,
    char * y )
```

TP n°: 2

Titre du TP : Merge Join

Date : 13/10/17

Nom : Lefranc Prenom : Joaquim email : [lefrancjoaquim@gmail.com](mailto:lefrancjoaquim@gmail.com)

Nom : Skoda Prenom : Jérôme email : [contact@jeromeskoda.fr](mailto:contact@jeromeskoda.fr)

## 5.6 Référence du fichier src/bdd/quicksort.h

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

### Fonctions

- void [quicksort](#) (char \*list, char m, char n)

### 5.6.1 Documentation des fonctions

#### 5.6.1.1 quicksort()

```
void quicksort (
    char * list,
    char m,
    char n )
```

TP n°: 2

Titre du TP : Merge Join

Date : 13/10/17

Nom : Lefranc Prenom : Joaquim email : [lefrancjoaquim@gmail.com](mailto:lefrancjoaquim@gmail.com)

Nom : Skoda Prenom : Jérôme email : [contact@jeromeskoda.fr](mailto:contact@jeromeskoda.fr)

Remarques : Ce fichier a été récupéré du site: <http://www.zentut.com/c-tutorial/c-quicksort-algorithm/>  
il a été réadapté dans le cadre du projet

## 5.7 Référence du fichier src/demo/1-natural-join.c

```
#include "../bdd/bdd.h"
```

Graphe des dépendances par inclusion de 1-natural-join.c: