

Projeto C LI3 - Sistema de Gestão de Vendas

Generated by Doxygen 1.8.17

1 File Index	1
1.1 File List	1
2 File Documentation	3
2.1 clients.h File Reference	3
2.1.1 Detailed Description	4
2.1.2 Typedef Documentation	4
2.1.2.1 CLIENTS	4
2.1.3 Function Documentation	4
2.1.3.1 addClient()	4
2.1.3.2 changeBranchClients()	4
2.1.3.3 destroyClients()	5
2.1.3.4 findClientesOfAllBranches()	5
2.1.3.5 findClientsNeverBoughtCount()	5
2.1.3.6 initClients()	6
2.1.3.7 initCLS()	6
2.1.3.8 listOfClients()	6
2.1.3.9 numberOfClientsNeverBought()	6
2.1.3.10 numberOfClientsOfAllBranches()	7
2.1.3.11 sortByLetter()	7
2.2 files.h File Reference	8
2.2.1 Detailed Description	8
2.2.2 Function Documentation	8
2.2.2.1 getFiles()	8
2.2.2.2 verifyFiles()	9
2.3 gets.h File Reference	9
2.3.1 Detailed Description	9
2.3.2 Function Documentation	10
2.3.2.1 getBranch()	10
2.3.2.2 getClient()	10
2.3.2.3 getLetter()	10
2.3.2.4 getMaxMonth()	10
2.3.2.5 getMonth()	11
2.3.2.6 getNumber()	11
2.3.2.7 getOneBranch()	11
2.3.2.8 getProduct()	11
2.4 interface.h File Reference	12
2.4.1 Detailed Description	13
2.4.2 Function Documentation	13
2.4.2.1 destroySGV()	13
2.4.2.2 getClientFavoriteProducts()	13
2.4.2.3 getClientsAndProductsNeverBoughtCount()	13

2.4.2.4	getClientsOfAllBranches()	14
2.4.2.5	getClientTopProfitProducts()	14
2.4.2.6	getCurrentFilesInfo()	14
2.4.2.7	getProductSalesAndProfit()	14
2.4.2.8	getProductsBoughtByClient()	16
2.4.2.9	getProductsBuyers()	16
2.4.2.10	getProductsNeverBought()	16
2.4.2.11	getProductsStartedByLetter()	17
2.4.2.12	getSalesAndProfit()	17
2.4.2.13	getTopSelledProducts()	17
2.4.2.14	initSGV()	18
2.4.2.15	loadSGVFromFiles()	18
2.4.2.16	showFilesInfo()	18
2.5	menu.h File Reference	19
2.5.1	Detailed Description	20
2.5.2	Function Documentation	20
2.5.2.1	clienteMesPalavras()	21
2.5.2.2	clientePalavra()	21
2.5.2.3	clientesQuery9()	21
2.5.2.4	dadosClientesMes()	21
2.5.2.5	dadosCompras()	21
2.5.2.6	dadosComprasG()	21
2.5.2.7	dadosQuery10()	22
2.5.2.8	dadosQuery11()	22
2.5.2.9	dadosQuery12()	22
2.5.2.10	dadosTotalVendas()	22
2.5.2.11	dataPalavra()	22
2.5.2.12	enterToContinue()	22
2.5.2.13	fatUniPalavras()	23
2.5.2.14	FatUniPalavrasG()	23
2.5.2.15	filesInfo()	23
2.5.2.16	filiaisPalavras()	23
2.5.2.17	getBranches2()	23
2.5.2.18	getClient2()	23
2.5.2.19	getClientAndMonth2()	23
2.5.2.20	getClientAndNumber()	24
2.5.2.21	getEnter()	24
2.5.2.22	getLetter2()	24
2.5.2.23	getMinAndMaxMonth2()	24
2.5.2.24	getNumber2()	24
2.5.2.25	getProductAndBranch2()	24
2.5.2.26	getProductAndMonth2()	24

2.5.2.27 infoPalavra()	25
2.5.2.28 letrasSGV()	25
2.5.2.29 loading()	25
2.5.2.30 menuPalavra()	25
2.5.2.31 miniMenuQuery2()	25
2.5.2.32 miniMenuQuery3()	25
2.5.2.33 miniMenuQuery5()	25
2.5.2.34 miniMenuQuery6()	26
2.5.2.35 miniMenuQuery9()	26
2.5.2.36 mostraClientesPalavras()	26
2.5.2.37 mostraClientesQuery5()	26
2.5.2.38 mostraContador()	26
2.5.2.39 mostraFilial()	26
2.5.2.40 mostraLetra()	27
2.5.2.41 mostraProdutosQuery2()	27
2.5.2.42 parteDeBaixoMenu()	27
2.5.2.43 parteDeBaixoTabela()	27
2.5.2.44 parteDeBaixoTempos()	27
2.5.2.45 parteDeCimaMenu()	27
2.5.2.46 parteDeCimaTabela()	27
2.5.2.47 parteDeCimaTempos()	28
2.5.2.48 printaDadosTotalVendas()	28
2.5.2.49 produtoFilialPalavras()	28
2.5.2.50 produtoMesPalavras()	28
2.5.2.51 query11Palavras()	28
2.5.2.52 setasPalavras()	28
2.5.2.53 showAndGetOptionsMenu2()	28
2.5.2.54 showClientFavoriteProducts()	29
2.5.2.55 showClientsAndProductsNeverBoughtCount()	29
2.5.2.56 showClientsOfAllBranches()	29
2.5.2.57 showClientTopProfitProducts()	29
2.5.2.58 showDestructionTime()	29
2.5.2.59 showMenu1()	30
2.5.2.60 showMenu2()	30
2.5.2.61 showMenuOption()	30
2.5.2.62 showProductSalesAndProfit()	30
2.5.2.63 showProductsBoughtByClient()	30
2.5.2.64 showProductsBuyers()	30
2.5.2.65 showProductsNeverBought()	31
2.5.2.66 showProductsStartbyLetter()	31
2.5.2.67 showSalesAndProfit()	31
2.5.2.68 showTopSoldProducts()	31

2.5.2.69 visaoGlobalPalvra()	31
2.6 products.h File Reference	31
2.6.1 Detailed Description	33
2.6.2 Typedef Documentation	33
2.6.2.1 PRODUCTS	33
2.6.2.2 PRODUCTSEARCH	33
2.6.3 Function Documentation	33
2.6.3.1 addProduct()	33
2.6.3.2 changeBranchProducts()	33
2.6.3.3 destroyProducts()	34
2.6.3.4 destroyPS()	34
2.6.3.5 findProductsByLetter()	34
2.6.3.6 findProductsNeverBought()	35
2.6.3.7 findProductsNeverBoughtCount()	35
2.6.3.8 getProductsOfPS()	36
2.6.3.9 getTotalOfPS()	36
2.6.3.10 initProducts()	36
2.6.3.11 initProductSearch()	36
2.6.3.12 iter_all()	37
2.6.3.13 listOfProducts()	37
2.6.3.14 numberOfProductsNeverBought()	38
2.6.3.15 productsNeverBought()	38
2.6.3.16 productsNeverBought0()	38
2.6.3.17 productsNeverBought1()	39
2.6.3.18 productsNeverBought2()	39
2.6.3.19 sizeOfProductsNeverBoughtAllBranches()	40
2.6.3.20 sizeOfProductsNeverBoughtEveryBranch()	40
2.7 sales.h File Reference	40
2.7.1 Detailed Description	42
2.7.2 Typedef Documentation	42
2.7.2.1 CLPROD	42
2.7.2.2 CLSEARCHER	42
2.7.2.3 PRODCL	43
2.7.2.4 PRSEARCHER	43
2.7.2.5 SALES	43
2.7.3 Function Documentation	43
2.7.3.1 addBranch()	43
2.7.3.2 changeClientsProducts()	44
2.7.3.3 changeProductsClients()	44
2.7.3.4 changeTotalProducts()	44
2.7.3.5 destroyClientsProducts()	45
2.7.3.6 destroyCLS()	45

2.7.3.7 destroyProductsClients()	45
2.7.3.8 destroySales()	46
2.7.3.9 findClientFavotiteProducts()	46
2.7.3.10 findCLS()	46
2.7.3.11 findData()	47
2.7.3.12 findProductSalesAndProfit()	47
2.7.3.13 findProductSellesByClient()	47
2.7.3.14 findSalesMerge()	49
2.7.3.15 getClientsAndUnitsCLS()	49
2.7.3.16 getCLSearcher()	50
2.7.3.17 getProductsAndUnitsPRS()	50
2.7.3.18 initClientProducts()	50
2.7.3.19 initProductsClients()	51
2.7.3.20 initSales()	51
2.7.3.21 mergeBranches()	51
2.7.3.22 mergeBranchesAndMonths()	52
2.7.3.23 numberOfSales()	52
2.7.3.24 sizeOfProducts()	52
2.7.3.25 sort()	53
2.8 sgv.h File Reference	53
2.8.1 Detailed Description	54
2.8.2 Typedef Documentation	54
2.8.2.1 SGV	55
2.8.3 Function Documentation	55
2.8.3.1 changeTotalMonth()	55
2.8.3.2 destructionSGV()	55
2.8.3.3 findSalesAndProfit()	55
2.8.3.4 getArrayProducts()	56
2.8.3.5 getArrayProductsByLetter()	56
2.8.3.6 getArraySales()	57
2.8.3.7 getClients()	57
2.8.3.8 getLinesValidatedAndRead()	57
2.8.3.9 getSales()	58
2.8.3.10 initialize_SGV()	58
2.8.3.11 loadSGV()	58
2.8.3.12 readClients()	59
2.8.3.13 readProducts()	59
2.8.3.14 readSales()	59
2.8.3.15 validadeSale()	60
2.8.3.16 validateBranch()	60
2.8.3.17 validateClient()	60
2.8.3.18 validateMonth()	60

2.8.3.19 validatePrice()	62
2.8.3.20 validateProduct()	62
2.8.3.21 validateType()	62
2.8.3.22 validateUnits()	63

Index	65
--------------	-----------

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

clients.h	Modulo que faz o tratamento dos clientes	3
files.h	Modulo de tratamento dos ficheiros	8
gets.h	Modulo de tratamento de interaçao com o utilizador	9
interface.h	Modulo que faz o tratamento das queries	12
menu.h	Modulo de tratamento do menu	19
products.h	Modulo que faz o tratamento dos produtos	31
sales.h	Modulo que faz o tratamento das vendas	40
sgv.h	Modulo de tratamento da Base de Dados	53

Chapter 2

File Documentation

2.1 clients.h File Reference

Modulo que faz o tratamento dos clientes.

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

Typedefs

- typedef struct clients * **CLIENTS**
Estrutura dos Clientes.

Functions

- **CLIENTS** **initClients** ()
Inicializador da Estrutura dos Clientes.
- void **destroyClients** (**CLIENTS** clients)
Destruição da Estrutura dos Clientes.
- void **addClient** (**CLIENTS** clients, char *client)
Adiciona um cliente a estrutura dos Clientes.
- void **changeBranchClients** (**CLIENTS** clients, char *client, int branch)
Altera valor do array branch, na estrutura Cliente, para que este esteja marcado como comprou um produto num filial.
- int **listOfClients** (**CLIENTS** clients, char *client)
Verifica se um cliente esta na Estrutura de Clientes.
- char ** **findClientesOfAllBranches** (**CLIENTS** clients, int *size)
Função que vai buscar todos os clientes que efetuaram compras em todas as filias e ordena-os.
- int **sortByLetter** (const void *a, const void *b)
Função auxiliar que ordena clientes.
- char ** **initCLS** (int size)
Função que inicializa um lista de strings.
- void **numberOfClientsOfAllBranches** (gpointer key, gpointer value, gpointer data)
Função auxiliar que conta o numero de clientes que efetuaram compras em todas as filias.
- int **findClientsNeverBoughtCount** (**CLIENTS** clients)
Função que conta o numero de clientes que nunca compraram.
- void **numberOfClientsNeverBought** (gpointer key, gpointer value, gpointer data)
Função auxiliar que conta o numero de clientes que nunca compraram.

2.1.1 Detailed Description

Modulo que faz o tratamento dos clientes.

2.1.2 Typedef Documentation

2.1.2.1 CLIENTS

```
typedef struct clients* CLIENTS
```

Estrutura dos Clientes.

2.1.3 Function Documentation

2.1.3.1 addClient()

```
void addClient (
    CLIENTS clients,
    char * client )
```

Adiciona um cliente a estrutura dos Clientes.

Parameters

<i>Estrutura</i>	a dos clientes
<i>Cliente</i>	a ser introduzido

2.1.3.2 changeBranchClients()

```
void changeBranchClients (
    CLIENTS clients,
    char * client,
    int branch )
```

Altera valor do array branch, na estrutura Cliente, para que este esteja marcado como comprou um produto num filial.

Parameters

<i>Estrutura</i>	a dos clientes
<i>Cliente</i>	a ser alterado
<i>Branch</i>	a ser alterada

2.1.3.3 destroyClients()

```
void destroyClients (
    CLIENTS clients )
```

Destruição da Estrutura dos Clientes.

Parameters

<i>Estrutura</i>	a ser destruída
------------------	-----------------

2.1.3.4 findClientesOfAllBranches()

```
char** findClientesOfAllBranches (
    CLIENTS clients,
    int * size )
```

Função que vai buscar todos os clientes que efetuaram compras em todas as filias e ordena-os.

Parameters

<i>Estrutura</i>	dos clientes
<i>Apontador</i>	do tamanho do numero de clientes

2.1.3.5 findClientsNeverBoughtCount()

```
int findClientsNeverBoughtCount (
    CLIENTS clients )
```

Função que conta o numero de clientes que nunca compraram.

Parameters

<i>Estrutura</i>	dos Clientes
------------------	--------------

Returns

Numero de clientes que nunca compraram

2.1.3.6 initClients()

```
CLIENTS initClients ( )
```

Inicializador da Estrutura dos Clientes.

Returns

Estrutura dos Clientes

2.1.3.7 initCLS()

```
char** initCLS (
    int size )
```

Função que inicializa um lista de strings.

Parameters

<i>Tamanho</i>	da lista
----------------	----------

Returns

Lista de strings inicializada

2.1.3.8 listOfClients()

```
int listOfClients (
    CLIENTS clients,
    char * client )
```

Verifica se um cliente esta na Estrutura de Clientes.

Parameters

<i>Estrutura</i>	a dos clientes
<i>Cliente</i>	a ser verificado

2.1.3.9 numberOfClientsNeverBought()

```
void numberOfClientsNeverBought (
    gpointer key,
```

```
gpointer value,  
gpointer data )
```

Função auxiliar que conta o numero de clientes que nunca compraram.

Parameters

<i>Key</i>	correspondente ao cliente
<i>Value</i>	correspondente ao cliente
<i>Data</i>	intrudozida (neste caso o numero de clientes que nunca compraram)

2.1.3.10 numberOfClientsOfAllBranches()

```
void numberOfClientsOfAllBranches (  
    gpointer key,  
    gpointer value,  
    gpointer data )
```

Função auxiliar que conta o numero de clientes que efetuaram compras em todas as filias.

Parameters

<i>Key</i>	correspondente ao cliente
<i>Value</i>	correspondente ao cliente
<i>Data</i>	intrudozida (neste caso o tamanho que ir tomar a lista de strings)

2.1.3.11 sortByLetter()

```
int sortByLetter (  
    const void * a,  
    const void * b )
```

Função auxiliar que ordena clientes.

Parameters

<i>Cliente</i>	1
<i>Cliente</i>	2

Returns

Retorna (0) caso sejam igual, (1) se cliente 2 > cliente 1 e (-1) de cliente 2 < cliente 1

2.2 files.h File Reference

Modulo de tratamento dos ficheiros.

```
#include <stdio.h>
#include <string.h>
#include "defines.h"
```

Functions

- void [getFiles](#) (char *clientsFilePath, char *productsFilePath, char *salesFilePath)
Busca os ficheiros a serem lidos.
- int [verifyFiles](#) (char *files, char *filePath)
Verifica se o ficheiro é valido.

2.2.1 Detailed Description

Modulo de tratamento dos ficheiros.

2.2.2 Function Documentation

2.2.2.1 getFiles()

```
void getFiles (
    char * clientsFilePath,
    char * productsFilePath,
    char * salesFilePath )
```

Busca os ficheiros a serem lidos.

Parameters

<i>Ficheiro</i>	dos Clientes
<i>Ficheiro</i>	dos Produtos
<i>Ficheiro</i>	das Vendas

2.2.2.2 verifyFiles()

```
int verifyFiles (
    char * files,
    char * filePath )
```

Verifica se o ficheiro é valido.

Parameters

<i>Ficheiro</i>	a ser testado
<i>String</i>	que guarda o file caso seja valido

2.3 gets.h File Reference

Modulo de tratamento de interação com o utilizador.

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include "sgv.h"
#include "defines.h"
#include "menu.h"
```

Functions

- char [getLetter](#) ()
Função que pede uma letra ao utilizador.
- void [getProduct](#) (SGV sgv, char product[])
 - Função que pede uma produto ao utilizador.*
- int [getMonth](#) ()
Função que pede um mes ao utilizador.
- int [getBranch](#) ()
Função que pede uma filial ao utilizador.
- void [getClient](#) (SGV sgv, char client[])
 - Função que pede uma cliente ao utilizador.*
- int [getMaxMonth](#) ()
Funcao que pede um mes ao utilizador para fazer um intervalo.
- int [getOneBranch](#) ()
Função que pede uma filial ao utilizador.
- int [getNumber](#) ()
Função que pede um numero ao utilizador.

2.3.1 Detailed Description

Modulo de tratamento de interação com o utilizador.

2.3.2 Function Documentation

2.3.2.1 getBranch()

```
int getBranch ( )
```

Função que pede uma filial ao utilizador.

Returns

Filial introduzido

2.3.2.2 getClient()

```
void getClient (
    SGV sgv,
    char client[] )
```

Função que pede uma cliente ao utilizador.

Parameters

<i>Estrutura</i>	de Dados para vereficar a existencia do cliente
<i>Cliente</i>	a ser devolvido

2.3.2.3 getLetter()

```
char getLetter ( )
```

Função que pede uma letra ao utilizador.

Returns

Letra a ser devolvida

2.3.2.4 getMaxMonth()

```
int getMaxMonth ( )
```

Funcao que pede um mes ao utilizador para fazer um intervalo.

Returns

Mes maximo

2.3.2.5 getMonth()

```
int getMonth ( )
```

Função que pede um mes ao utilizador.

Returns

Mes introduzido

2.3.2.6 getNumber()

```
int getNumber ( )
```

Função que pede um numero ao utilizador.

Returns

Numero introduzido

2.3.2.7 getOneBranch()

```
int getOneBranch ( )
```

Função que pede uma filial ao utilizador.

Returns

Filial introduzido

2.3.2.8 getProduct()

```
void getProduct (
    SGV sgv,
    char product[ ] )
```

Função que pede uma produto ao utilizador.

Parameters

<i>Estrutura</i>	de Dados para vereficar a existencia do produto
<i>Produto</i>	a ser devolvido

2.4 interface.h File Reference

Modulo que faz o tratamento das queries.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "sgv.h"
#include "defines.h"
#include "menu.h"
#include "files.h"
#include "gets.h"
#include "sales.h"
#include "products.h"
```

Functions

- [SGV initSGV](#) ()
Inicialização da Base de Dados.
- void [destroySGV](#) ([SGV](#) sgv)
Destruição da Base de Dados.
- [SGV loadSGVFromFiles](#) ([SGV](#) sgv, char *clientsFilePath, char *productsFilePath, char *salesFilePath)
Carregar Base de Dados (SGV) a partir de ficheiros.
- void [getCurrentFilesInfo](#) ([SGV](#) sgv)
- void [getProductsStartedByLetter](#) ([SGV](#) sgv, char letter)
Função que vai buscar todos produtos que começam por um letra (QUERY2)
- void [getProductSalesAndProfit](#) ([SGV](#) sgv, char *productID, int month)
Função que vai buscar todas as vendas e faturação ocorridas num certo mes por um produto(QUERY3)
- void [getProductsNeverBought](#) ([SGV](#) sgv, int branchID)
Função que vai buscar todos os produtos que nao foram vendidos (por filiais ou geral) (QUERY4)
- void [getClientsOfAllBranches](#) ([SGV](#) sgv)
Função que vai buscar todos os clientes que efetuaram compras em todas as filias (QUERY5)
- void [getClientsAndProductsNeverBoughtCount](#) ([SGV](#) sgv)
Função que vai calcular o numero de produtos e clientes que nao efetuaram compras (QUERY6)
- void [getProductsBoughtByClient](#) ([SGV](#) sgv, char *clientID)
Função que vai buscar o numero de compras de um determinado cliente (QUERY7)
- void [getSalesAndProfit](#) ([SGV](#) sgv, int minMonth, int maxMonth)
Função que vai buscar o numero de compras e a faturação num intervalo de meses (QUERY8)
- void [getProductsBuyers](#) ([SGV](#) sgv, char *productID, int branch)
Função que vai buscar os clientes que compraram um produto num certo filial (QUERY9)
- void [getClientFavoriteProducts](#) ([SGV](#) sgv, char *clientID, int month)
Função que vai buscar os produtos comprados por um cliente num certo mes, ordenando-os (QUERY10)
- void [getTopSelledProducts](#) ([SGV](#) sgv, int limit)
Função que vai buscar os produtos mais vendidos (QUERY11)
- void [getClientTopProfitProducts](#) ([SGV](#) sgv, char *clientID, int limit)
Função que encontra os 'n' produtos mais comprados pelo cliente (QUERY12)
- void [showFilesInfo](#) ([SGV](#) sgv, char *clientsFilePath, char *productsFilePath, char *salesFilePath, float time)
Função que mostra a informação sobre a leitura dos files da primeira query (QUERY13)

2.4.1 Detailed Description

Modulo que faz o tratamento das queries.

2.4.2 Function Documentation

2.4.2.1 destroySGV()

```
void destroySGV (
    SGV sgv )
```

Destrução da Base de Dados.

Parameters

<i>Base</i>	de Dados a ser destruida
-------------	--------------------------

2.4.2.2 getClientFavoriteProducts()

```
void getClientFavoriteProducts (
    SGV sgv,
    char * clientID,
    int month )
```

Função que vai buscar os produtos comprados por um cliente num certo mes, ordenando-os (QUERY10)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Cliente</i>	a ser procurado
<i>Mes</i>	a ser procurada

2.4.2.3 getClientsAndProductsNeverBoughtCount()

```
void getClientsAndProductsNeverBoughtCount (
    SGV sgv )
```

Função que vai calcular o numero de produtos e clientes que nao efetuaram compras (QUERY6)

Parameters

<i>Estrutura</i>	de Dados (SGV)
------------------	----------------

2.4.2.4 getClientOfAllBranches()

```
void getClientOfAllBranches (
    SGV sgv )
```

Função que vai buscar todos os clientes que efetuaram compras em todas as filias (QUERY5)

Parameters

<i>Estrutura</i>	de Dados (SGV)
------------------	----------------

2.4.2.5 getClientTopProfitProducts()

```
void getClientTopProfitProducts (
    SGV sgv,
    char * clientID,
    int limit )
```

Função que encontra os 'n' produtos mais comprados pelo cliente (QUERY12)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Cliente</i>	em questao
<i>Numero</i>	de produtos a pesquisar

2.4.2.6 getCurrentFilesInfo()

```
void getCurrentFilesInfo (
    SGV sgv )
```

2.4.2.7 getProductSalesAndProfit()

```
void getProductSalesAndProfit (
    SGV sgv,
```

```
char * productID,  
int month )
```

Função que vai buscar todas as vendas e faturação ocorridas num certo mes por um produto(QUERY3)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Produto</i>	a ser examinado
<i>Mes</i>	em questao

2.4.2.8 getProductsBoughtByClient()

```
void getProductsBoughtByClient (
    SGV sgv,
    char * clientID )
```

Função que vai buscar o numero de compras de um determinado cliente (QUERY7)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Cliente</i>	a ser examinado

2.4.2.9 getProductsBuyers()

```
void getProductsBuyers (
    SGV sgv,
    char * productID,
    int branch )
```

Função que vai buscar os clientes que compraram um produto num certo filial (QUERY9)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Produto</i>	a ser procurado
<i>Filial</i>	a ser procurada

2.4.2.10 getProductsNeverBought()

```
void getProductsNeverBought (
    SGV sgv,
    int branchID )
```

Função que vai buscar todos os produtos que nao foram vendidos (por filiais ou geral) (QUERY4)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Filial</i>	a procurar produtos

2.4.2.11 getProductsStartedByLetter()

```
void getProductsStartedByLetter (
    SGV sgv,
    char letter )
```

Função que vai buscar todos produtos que começam por um letra (QUERY2)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Letra</i>	inicial do produto

2.4.2.12 getSalesAndProfit()

```
void getSalesAndProfit (
    SGV sgv,
    int minMonth,
    int maxMonth )
```

Função que vai buscar o numero de compras e a faturação num intervalo de meses (QUERY8)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Mes</i>	minimo
<i>Mes</i>	maximo

2.4.2.13 getTopSelledProducts()

```
void getTopSelledProducts (
    SGV sgv,
    int limit )
```

Função que vai buscar os produtos mais vendidos (QUERY11)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Numero</i>	de produtos a pesquisar

2.4.2.14 initSGV()

```
SGV initSGV ( )
```

Inicialização da Base de Dados.

Returns

Base de Dados

2.4.2.15 loadSGVFromFiles()

```
SGV loadSGVFromFiles (
    SGV sgv,
    char * clientsFilePath,
    char * productsFilePath,
    char * salesFilePath )
```

Carregar Base de Dados (SGV) a partir de ficheiros.

Parameters

<i>Ficheiro</i>	dos Clientes
<i>Ficheiro</i>	dos Produtos
<i>Ficheiro</i>	das Vendas

Returns

Base de Dados (SGV) carregada

2.4.2.16 showFilesInfo()

```
void showFilesInfo (
    SGV sgv,
    char * clientsFilePath,
    char * productsFilePath,
    char * salesFilePath,
    float time )
```

Função que mostra a informação sobre a leitura dos files da primeira query (QUERY13)

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Ficheiro</i>	dos clientes
<i>Ficheiro</i>	dos produtos
<i>Ficheiro</i>	das vendas
<i>Tempo</i>	de construção da base de dados

2.5 menu.h File Reference

Modulo de tratamento do menu.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include "defines.h"
#include "sgv.h"
#include "products.h"
#include "clients.h"
```

Functions

- int [showMenuOption](#) ()
- int [showMenu1](#) (int valid)
- int [showAndGetOptionMenu2](#) (int valid)
- void [showMenu2](#) (int op)
- void [parteDeCimaMenu](#) ()
- void [parteDeBaixoMenu](#) ()
- void [letrasSGV](#) ()
- void [menuPalavra](#) ()
- void [loading](#) ()
- void [parteDeCimaTabela](#) ()
- void [parteDeBaixoTabela](#) ()
- void [infoPalavra](#) ()
- void [dataPalavra](#) ()
- void [parteDeCimaTempos](#) ()
- void [parteDeBaixoTempos](#) ()
- void [getEnter](#) ()
- void [enterToContinue](#) ()
- void [mostraLetra](#) (char letter)
- void [mostraProdutosQuery2](#) (char **prods, int size, int nPages, int pag)
- void [miniMenuQuery2](#) (float time, int nPages, int pag, int size)
- void [produtoMesPalavras](#) (char *product, int month)
- void [filiaisPalavras](#) ()
- void [fatUniPalavras](#) ()
- void [dadosCompras](#) (double res[][4])
- void [visaoGlobalPalvra](#) ()
- void [FatUniPalavrasG](#) ()
- void [dadosComprasG](#) (double res[][4])

- void [miniMenuQuery3](#) (float time, int op)
- void [mostraFilial](#) (int filial)
- void [mostraClientesQuery5](#) (char **clients, int size, int nPages, int pag)
- void [setasPalavras](#) ()
- void [miniMenuQuery5](#) (float time, int size, int nPages, int pag)
- void [mostraContador](#) (int res[])
- void [miniMenuQuery6](#) (float time)
- void [clientePalavra](#) (char *client)
- void [dadosClientesMes](#) (int res[][12])
- void [dadosTotalVendas](#) (int *intervalo, double *res)
- void [printaDadosTotalVendas](#) (double res, int q)
- void [produtoFilialPalavras](#) (char *produto, int filial)
- void [clientesQuery9](#) (char **clientesN, int unidN, char **clientesP, int unidP, int maior)
- void [miniMenuQuery9](#) (int total, float time)
- void [clienteMesPalavras](#) (char *client, int month)
- void [dadosQuery10](#) (char **prods, int qt[], int size)
- void [query11Palavras](#) ()
- void [dadosQuery11](#) (char **prods, int data[][3][2], int size, int nPages, int pag)
- void [mostraClientesPalavras](#) (char *client)
- void [dadosQuery12](#) (char **prods, double qt[], int limit, int pag)
- void [showProductsStartbyLetter](#) (char **prods, char letter, int size, float time)
- void [showProductSalesAndProfit](#) (char *product, int month, double res[][4], float time)
- void [showProductsNeverBought](#) (char **prods, int size, int branchID, float time)
- void [showClientsOfAllBranches](#) (char **clients, int size, float time)
- void [showClientsAndProductsNeverBoughtCount](#) (int res[], float time)
- void [showProductsBoughtByClient](#) (char *client, int res[][12], float time)
- void [showSalesAndProfit](#) (int minMonth, int maxMonth, double res[], float time)
- void [showProductsBuyers](#) (char *productID, int branch, char **clientsN, char **clientsP, int unitsN, int unitsP, float time)
- void [showClientFavoriteProducts](#) (char **prods, int qt[], char *client, int month, int size, float time)
- void [showTopSoldProducts](#) (char **prods, int data[][3][2], int limit, float time)
- void [showClientTopProfitProducts](#) (char **prods, double qt[], char *client, int limit, float time)
- void [filesInfo](#) (int l[], char *clients, char *prods, char *sales, float time)
- void [showDestructionTime](#) (float time)
- char [getLetter2](#) ()
- void [getProductAndMonth2](#) (SGV sgv, char product[], int *month)
- int [getBranches2](#) ()
- void [getClient2](#) (SGV sgv, char client[])
- void [getMinAndMaxMonth2](#) (int *minMonth, int *maxMonth)
- void [getProductAndBranch2](#) (SGV sgv, char *product, int *branch)
- void [getClientAndMonth2](#) (SGV sgv, char *client, int *month)
- int [getNumber2](#) ()
- int [getClientAndNumber](#) (SGV sgv, char *client)

2.5.1 Detailed Description

Modulo de tratamento do menu.

2.5.2 Function Documentation

2.5.2.1 clienteMesPalavras()

```
void clienteMesPalavras (
    char * client,
    int month )
```

2.5.2.2 clientePalavra()

```
void clientePalavra (
    char * client )
```

2.5.2.3 clientesQuery9()

```
void clientesQuery9 (
    char ** clientesN,
    int unidN,
    char ** clientesP,
    int unidP,
    int maior )
```

2.5.2.4 dadosClientesMes()

```
void dadosClientesMes (
    int res[][12] )
```

2.5.2.5 dadosCompras()

```
void dadosCompras (
    double res[][4] )
```

2.5.2.6 dadosComprasG()

```
void dadosComprasG (
    double res[][4] )
```

2.5.2.7 dadosQuery10()

```
void dadosQuery10 (
    char ** prods,
    int qt[],
    int size )
```

2.5.2.8 dadosQuery11()

```
void dadosQuery11 (
    char ** prods,
    int data[][3][2],
    int size,
    int nPages,
    int pag )
```

2.5.2.9 dadosQuery12()

```
void dadosQuery12 (
    char ** prods,
    double qt[],
    int limit,
    int pag )
```

2.5.2.10 dadosTotalVendas()

```
void dadosTotalVendas (
    int * intervalo,
    double * res )
```

2.5.2.11 dataPalavra()

```
void dataPalavra ( )
```

2.5.2.12 enterToContinue()

```
void enterToContinue ( )
```

2.5.2.13 fatUniPalavras()

```
void fatUniPalavras ( )
```

2.5.2.14 FatUniPalavrasG()

```
void FatUniPalavrasG ( )
```

2.5.2.15 filesInfo()

```
void filesInfo (
    int l[],
    char * clients,
    char * prods,
    char * sales,
    float time )
```

2.5.2.16 filiaisPalavras()

```
void filiaisPalavras ( )
```

2.5.2.17 getBranches2()

```
int getBranches2 ( )
```

2.5.2.18 getClient2()

```
void getClient2 (
    SGV sgv,
    char client[] )
```

2.5.2.19 getClientAndMonth2()

```
void getClientAndMonth2 (
    SGV sgv,
    char * client,
    int * month )
```

2.5.2.20 getClientAndNumber()

```
int getClientAndNumber (
    SGV sgv,
    char * client )
```

2.5.2.21 getEnter()

```
void getEnter ( )
```

2.5.2.22 getLetter2()

```
char getLetter2 ( )
```

2.5.2.23 getMinAndMaxMonth2()

```
void getMinAndMaxMonth2 (
    int * minMonth,
    int * maxMonth )
```

2.5.2.24 getNumber2()

```
int getNumber2 ( )
```

2.5.2.25 getProductAndBranch2()

```
void getProductAndBranch2 (
    SGV sgv,
    char * product,
    int * branch )
```

2.5.2.26 getProductAndMonth2()

```
void getProductAndMonth2 (
    SGV sgv,
    char product[],
    int * month )
```


2.5.2.27 infoPalavra()

```
void infoPalavra ( )
```

2.5.2.28 letrasSGV()

```
void letrasSGV ( )
```

2.5.2.29 loading()

```
void loading ( )
```

2.5.2.30 menuPalavra()

```
void menuPalavra ( )
```

2.5.2.31 miniMenuQuery2()

```
void miniMenuQuery2 (
    float time,
    int nPages,
    int pag,
    int size )
```

2.5.2.32 miniMenuQuery3()

```
void miniMenuQuery3 (
    float time,
    int op )
```

2.5.2.33 miniMenuQuery5()

```
void miniMenuQuery5 (
    float time,
    int size,
    int nPages,
    int pag )
```

2.5.2.34 miniMenuQuery6()

```
void miniMenuQuery6 (
    float time )
```

2.5.2.35 miniMenuQuery9()

```
void miniMenuQuery9 (
    int total,
    float time )
```

2.5.2.36 mostraClientesPalavras()

```
void mostraClientesPalavras (
    char * client )
```

2.5.2.37 mostraClientesQuery5()

```
void mostraClientesQuery5 (
    char ** clients,
    int size,
    int nPages,
    int pag )
```

2.5.2.38 mostraContador()

```
void mostraContador (
    int res[] )
```

2.5.2.39 mostraFilial()

```
void mostraFilial (
    int filial )
```

2.5.2.40 mostraLetra()

```
void mostraLetra (
    char letter )
```

2.5.2.41 mostraProdutosQuery2()

```
void mostraProdutosQuery2 (
    char ** prods,
    int size,
    int nPages,
    int pag )
```

2.5.2.42 parteDeBaixoMenu()

```
void parteDeBaixoMenu ( )
```

2.5.2.43 parteDeBaixoTabela()

```
void parteDeBaixoTabela ( )
```

2.5.2.44 parteDeBaixoTempos()

```
void parteDeBaixoTempos ( )
```

2.5.2.45 parteDeCimaMenu()

```
void parteDeCimaMenu ( )
```

2.5.2.46 parteDeCimaTabela()

```
void parteDeCimaTabela ( )
```

2.5.2.47 parteDeCimaTempos()

```
void parteDeCimaTempos ( )
```

2.5.2.48 printaDadosTotalVendas()

```
void printaDadosTotalVendas (
    double res,
    int q )
```

2.5.2.49 produtoFilialPalavras()

```
void produtoFilialPalavras (
    char * produto,
    int filial )
```

2.5.2.50 produtoMesPalavras()

```
void produtoMesPalavras (
    char * product,
    int month )
```

2.5.2.51 query11Palavras()

```
void query11Palavras ( )
```

2.5.2.52 setasPalavras()

```
void setasPalavras ( )
```

2.5.2.53 showAndGetOptionsMenu2()

```
int showAndGetOptionsMenu2 (
    int valid )
```

2.5.2.54 showClientFavoriteProducts()

```
void showClientFavoriteProducts (
    char ** prods,
    int qt[],
    char * client,
    int month,
    int size,
    float time )
```

2.5.2.55 showClientsAndProductsNeverBoughtCount()

```
void showClientsAndProductsNeverBoughtCount (
    int res[],
    float time )
```

2.5.2.56 showClientsOfAllBranches()

```
void showClientsOfAllBranches (
    char ** clients,
    int size,
    float time )
```

2.5.2.57 showClientTopProfitProducts()

```
void showClientTopProfitProducts (
    char ** prods,
    double qt[],
    char * client,
    int limit,
    float time )
```

2.5.2.58 showDestructionTime()

```
void showDestructionTime (
    float time )
```

2.5.2.59 showMenu1()

```
int showMenu1 (
    int valid )
```

2.5.2.60 showMenu2()

```
void showMenu2 (
    int op )
```

2.5.2.61 showMenuOption()

```
int showMenuOption ( )
```

2.5.2.62 showProductSalesAndProfit()

```
void showProductSalesAndProfit (
    char * product,
    int month,
    double res[][4],
    float time )
```

2.5.2.63 showProductsBoughtByClient()

```
void showProductsBoughtByClient (
    char * client,
    int res[][12],
    float time )
```

2.5.2.64 showProductsBuyers()

```
void showProductsBuyers (
    char * productID,
    int branch,
    char ** clientsN,
    char ** clientsP,
    int unitsN,
    int unitsP,
    float time )
```

2.5.2.65 showProductsNeverBought()

```
void showProductsNeverBought (
    char ** prods,
    int size,
    int branchID,
    float time )
```

2.5.2.66 showProductsStartbyLetter()

```
void showProductsStartbyLetter (
    char ** prods,
    char letter,
    int size,
    float time )
```

2.5.2.67 showSalesAndProfit()

```
void showSalesAndProfit (
    int minMonth,
    int maxMonth,
    double res[],
    float time )
```

2.5.2.68 showTopSoldProducts()

```
void showTopSoldProducts (
    char ** prods,
    int data[][3][2],
    int limit,
    float time )
```

2.5.2.69 visaoGlobalPalvra()

```
void visaoGlobalPalvra ( )
```

2.6 products.h File Reference

Modulo que faz o tratamento dos produtos.

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

Typedefs

- typedef struct products * [PRODUCTS](#)
Estrutura dos Produtos.
- typedef struct productSearch * [PRODUCTSEARCH](#)
Estrutura que guarda os produtos durante uma procura.

Functions

- [PRODUCTS](#) [initProducts](#) ()
Inicializador da Estrutura dos Produtos.
- void [destroyProducts](#) ([PRODUCTS](#) products)
Destruicao da Estrutura dos Produtos.
- void [addProduct](#) ([PRODUCTS](#) products, char *product)
Adiciona um produto a estrutura dos Produtos.
- void [changeBranchProducts](#) ([PRODUCTS](#) products, char *product, int branch)
Altera value do produto, indicando em qual filial o produto foi vendido.
- int [listOfProducts](#) ([PRODUCTS](#) products, char *product)
Verifica se um produto esta na Estrutura de Produtos.
- [PRODUCTSEARCH](#) [findProductsByLetter](#) ([PRODUCTS](#) products)
Função que encontra os produtos começados por uma determinada letra.
- int [iter_all](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que atravessa todos os nodos de uma arvore.
- [PRODUCTSEARCH](#) [initProductSearch](#) (int size)
Inicializa uma estrutura de procura para os produtos.
- char ** [getProductsOfPS](#) ([PRODUCTSEARCH](#) ps)
Função que passa os produtos encontrados pela estrutura [PRODUCTSEARCH](#) para um lista de strings.
- int [getTotalOfPS](#) ([PRODUCTSEARCH](#) ps)
Função que retorna o numero de produtos encontrados pela estrutura [PRODUCTSEARCH](#).
- void [destroyPS](#) ([PRODUCTSEARCH](#) ps)
Função que destroi a estrutura [PRODUCTSEARCH](#).
- [PRODUCTSEARCH](#) [findProductsNeverBought](#) ([PRODUCTS](#) *products, int branch)
Função que encontra todos os produtos nao vendidos de uma ou todas filial.
- int [productsNeverBought0](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que atravessa todos os nodos de uma arvore e encontra os produtos nao vendidos na filial 1.
- int [productsNeverBought1](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que atravessa todos os nodos de uma arvore e encontra os produtos nao vendidos na filial 2.
- int [productsNeverBought2](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que atravessa todos os nodos de uma arvore e encontra os produtos nao vendidos na filial 3.
- int [sizeOfProductsNeverBoughtEveryBranch](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que atravessa todos os nodos de uma arvore e calcula o numero de produtos nao vendidos em todas a filiais.
- int [sizeOfProductsNeverBoughtAllBranches](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que atravessa todos os nodos de uma arvore e calcula o numero de produtos nao vendidos.
- int [productsNeverBought](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que atravessa todos os nodos de uma arvore e encontra os produtos nao vendidos em todas as filiais.
- int [findProductsNeverBoughtCount](#) ([PRODUCTS](#) *products)
Função que conta o numero de produtos que nunca foram comprados.
- int [numberOfProductsNeverBought](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que conta o numero de produtos que nunca foram comprados.

2.6.1 Detailed Description

Modulo que faz o tratamento dos produtos.

2.6.2 Typedef Documentation

2.6.2.1 PRODUCTS

```
typedef struct products* PRODUCTS
```

Estrutura dos Produtos.

2.6.2.2 PRODUCTSEARCH

```
typedef struct productSearch* PRODUCTSEARCH
```

Estrutura que guarda os produtos durante uma procura.

2.6.3 Function Documentation

2.6.3.1 addProduct()

```
void addProduct (
    PRODUCTS products,
    char * product )
```

Adiciona um produto a estrutura dos Produtos.

Parameters

<i>Estrutura</i>	a dos produtos
<i>Produto</i>	a ser introduzido

2.6.3.2 changeBranchProducts()

```
void changeBranchProducts (
    PRODUCTS products,
```

```
char * product,  
int branch )
```

Altera value do produto, indicando em qual filial o produto foi vendido.

Parameters

<i>Estrutura</i>	a dos produtos
<i>Produto</i>	a ser aleterado
<i>Filial</i>	em que a venda foi efetuada

2.6.3.3 destroyProducts()

```
void destroyProducts (  
    PRODUCTS products )
```

Destruição da Estrutura dos Produtos.

Parameters

<i>Estrutura</i>	a ser destruida
------------------	-----------------

2.6.3.4 destroyPS()

```
void destroyPS (  
    PRODUCTSEARCH ps )
```

Função que destroi a estrutura PRODUCTSEARCH.

Parameters

<i>Estrutura</i>	PRODUCTSEARCH a ser eliminada
------------------	-------------------------------

2.6.3.5 findProductsByLetter()

```
PRODUCTSEARCH findProductsByLetter (  
    PRODUCTS products )
```

Função que encontra os produtos começados por uma determinada letra.

Parameters

<i>Estrutura</i>	a dos produtos
------------------	----------------

Returns

Estrutura a qual guarda os produtos procurados

2.6.3.6 findProductsNeverBought()

```
PRODUCTSEARCH findProductsNeverBought (
    PRODUCTS * products,
    int branch )
```

Função que encontra todos os produtos nao vendidos de uma ou todas filial.

Parameters

<i>Array</i>	de Estrutura de Produtos
<i>Filial</i>	a procurar

Returns

Array de Estruturas PRODUCTSEARCH

2.6.3.7 findProductsNeverBoughtCount()

```
int findProductsNeverBoughtCount (
    PRODUCTS * products )
```

Função que conta o numero de produtos que nunca foram comprados.

Parameters

<i>Array</i>	de Estrutura dos Produtos
--------------	---------------------------

Returns

Numero de produtos que nunca foram comprados

2.6.3.8 getProductsOfPS()

```
char** getProductsOfPS (
    PRODUCTSEARCH ps )
```

Função que passa os produtos encontrados pela estrutura PRODUCTSEARCH para um lista de strings.

Parameters

<i>Estrutura</i>	PRODUCTSEARCH
------------------	---------------

Returns

Lista de Strings

2.6.3.9 getTotalOfPS()

```
int getTotalOfPS (
    PRODUCTSEARCH ps )
```

Função que retorna o numero de produtos encontrados pela estrutura PRODUCTSEARCH.

Parameters

<i>Estrutura</i>	PRODUCTSEARCH
------------------	---------------

Returns

Numero de produtos lidos

2.6.3.10 initProducts()

```
PRODUCTS initProducts ( )
```

Inicializador da Estrutura dos Produtos.

Returns

Estrutura dos Produtos

2.6.3.11 initProductSearch()

```
PRODUCTSEARCH initProductSearch (
    int size )
```

Inicializa uma estrutura de procura para os produtos.

Parameters

<i>Tamanho</i>	da lista a strings
----------------	--------------------

Returns

Estrutura inicializada

2.6.3.12 iter_all()

```
int iter_all (
    gpointer key,
    gpointer value,
    gpointer data )
```

Função auxiliar que atravessa todos os nodos de uma arvore.

Parameters

<i>Key</i>	correspondente ao nodo
<i>Value</i>	correspondente ao nodo
<i>Paramentro</i>	passado pela função "g_hash_table_foreach" (neste caso a estrutura PRODUCTSEARCH)

Returns

Inteiro caso queria parar a função

2.6.3.13 listOfProducts()

```
int listOfProducts (
    PRODUCTS products,
    char * product )
```

Verifica se um produto esta na Estrutura de Produtos.

Parameters

<i>Estrutura</i>	a dos produtos
<i>Produto</i>	a ser verificado

2.6.3.14 numberOfProductsNeverBought()

```
int numberOfProductsNeverBought (
    gpointer key,
    gpointer value,
    gpointer data )
```

Função auxiliar que conta o numero de produtos que nunca foram comprados.

Parameters

<i>Key</i>	correspondente ao produto
<i>Value</i>	correspondente ao produto
<i>Data</i>	intrudozida (neste caso o numero de produtos que nunca foram comprados)

Returns

Inteiro caso queria parar a função

2.6.3.15 productsNeverBought()

```
int productsNeverBought (
    gpointer key,
    gpointer value,
    gpointer data )
```

Função auxiliar que atravessa todos os nodos de uma arvore e encontra os produtos nao vendidos em todas as filiais.

Parameters

<i>Key</i>	correspondente ao nodo
<i>Value</i>	correspondente ao nodo
<i>Paramentro</i>	passado pela função "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a função

2.6.3.16 productsNeverBought0()

```
int productsNeverBought0 (
    gpointer key,
    gpointer value,
    gpointer data )
```

Função auxiliar que atravessa todos os nodos de uma arvore e encontra os produtos nao vendidos na filial 1.

Parameters

<i>Key</i>	correspondente ao nodo
<i>Value</i>	correspondente ao nodo
<i>Paramentro</i>	passado pela função "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a função

2.6.3.17 productsNeverBought1()

```
int productsNeverBought1 (  
    gpointer key,  
    gpointer value,  
    gpointer data )
```

Função auxiliar que atravessa todos os nodos de uma árvore e encontra os produtos não vendidos na filial 2.

Parameters

<i>Key</i>	correspondente ao nodo
<i>Value</i>	correspondente ao nodo
<i>Paramentro</i>	passado pela função "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a função

2.6.3.18 productsNeverBought2()

```
int productsNeverBought2 (  
    gpointer key,  
    gpointer value,  
    gpointer data )
```

Função auxiliar que atravessa todos os nodos de uma árvore e encontra os produtos não vendidos na filial 3.

Parameters

<i>Key</i>	correspondente ao nodo
<i>Value</i>	correspondente ao nodo
<i>Paramentro</i>	passado pela função "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a função

2.6.3.19 sizeofProductsNeverBoughtAllBranches()

```
int sizeofProductsNeverBoughtAllBranches (
    gpointer key,
    gpointer value,
    gpointer data )
```

Função auxiliar que atravessa todos os nodos de uma árvore e calcula o número de produtos não vendidos.

Parameters

<i>Key</i>	correspondente ao nodo
<i>Value</i>	correspondente ao nodo
<i>Paramentro</i>	passado pela função "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a função

2.6.3.20 sizeofProductsNeverBoughtEveryBranch()

```
int sizeofProductsNeverBoughtEveryBranch (
    gpointer key,
    gpointer value,
    gpointer data )
```

Função auxiliar que atravessa todos os nodos de uma árvore e calcula o número de produtos não vendidos em todas as filiais.

Parameters

<i>Key</i>	correspondente ao nodo
<i>Value</i>	correspondente ao nodo
<i>Paramentro</i>	passado pela função "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a função

2.7 sales.h File Reference

Módulo que faz o tratamento das vendas.


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

Typedefs

- typedef struct sales * [SALES](#)
Estrutura das vendas.
- typedef struct prodCl * [PRODCL](#)
Estrutura que relaciona produtos com clientes.
- typedef struct clProd * [CLPROD](#)
Estrutura que relaciona clientes com produtos.
- typedef struct clSearcher * [CLSEARCHER](#)
Estrutura que guarda clientes numa procura.
- typedef struct prSearcher * [PRSEARCHER](#)
Estrutura que guarda produtos numa procura.

Functions

- [SALES](#) [initSales](#) ()
Inicializador da Estrutura das Vendas.
- void [destroySales](#) ([SALES](#) sales)
Destrução da Estrutura das Vendas.
- void [destroyProductsClients](#) (void *prCl)
Destrução da Estrutura que relaciona produtos com clientes.
- void [destroyClientsProducts](#) (void *clPr)
Destrução da Estrutura que relaciona clientes com produtos.
- void [addBranch](#) ([SALES](#) sales, char *product, char *client, char type, float price, int units, int month)
Carrega uma venda a Estrutura de Vendas.
- [CLPROD](#) [initClientProducts](#) ()
Inicialização da Estrutura que relaciona Clientes com Produtos.
- void [changeClientsProducts](#) ([CLPROD](#) cp, char *product, int month, int units, float price)
Altera a Estrutura que relaciona clientes com produtos, alterando a faturação do produto.
- [PRODCL](#) [initProductsClients](#) ()
Inicialização da Estrutura que relaciona Produtos com Clientes.
- void [changeTotalProducts](#) (double total[][2][12], char type, float price, int month)
Altera a Estrutura que relaciona produtos com clientes, aumentando o preço e o numero de vendas sofridas por um produto.
- void [changeProductsClients](#) ([PRODCL](#) cl, char *client, char type)
Altera a Estrutura que relaciona produtos com clientes, alterando o tipo de venda que o cliente fez.
- void [findProductSalesAndProfit](#) ([SALES](#) *sales, char *productID, int month, double res[][4])
Função que encontra as vendas e a faturação de um produto num certo mes.
- void [findProductSellesByClient](#) ([SALES](#) sales, char *client, int *res)
Função que encontra as vendas feitas por um cliente.
- [CLSEARCHER](#) [getCLSearcher](#) ([SALES](#) sale, char *productID)
Função que encontra as clientes que compraram um produto.
- int [numberOfSales](#) ([PRODCL](#) pc, int flag)
Função que encontra o numero de vendas feitas.
- void [findCLS](#) (gpointer key, gpointer value, gpointer data)
Função auxiliar que encontra clientes.

- int [getClientAndUnitsCLS](#) ([CLSEARCHER](#) cls, char ***clientes, int flag)
Função que retira os clientes e as unidades da Estrutura de procura "CLSEARCHER".
- void [destroyCLS](#) ([CLSEARCHER](#) cls)
Função que elimina a estrutura de procura "CLSEARCHER".
- [PRSEARCHER](#) [findClientFavotiteProducts](#) ([SALES](#) *sales, char *client, int month, int *size)
Função que constroi a estrutura de procura "PRSEARCHER".
- int [sizeOfProducts](#) ([SALES](#) sales, char *client, int month)
Função que calcula o numero de produtos comprados por um cliente.
- int [sort](#) (const void *one, const void *two)
Função auxiliar que retorna se o segundo parametro é maior do que o primeiro.
- char ** [getProductsAndUnitsPRS](#) ([PRSEARCHER](#) prs, int size, int qt[])
Função encontra os produtos da estrutura "PRSEARCHER" e coloca-os numa lista de strings.
- GHashTable * [mergeBranches](#) ([SALES](#) *sales)
Função que junta as filiais numa unica hash table.
- char ** [findSalesMerge](#) (GHashTable *merge, double qt[], int size, int limit, int flag)
Função encontra os 'n' produtos que mais faturou em todo ano (este executa a query 11 e query a partir de uma flag)
- void [findData](#) ([SALES](#) *sales, int data[][3][2], char **prods, int limit, int branch)
- GHashTable * [mergeBranchesAndMonths](#) ([SALES](#) *sales, char *client)
Função que junta as filiais devididas em meses numa unica hash table.

2.7.1 Detailed Description

Modulo que faz o tratamento das vendas.

2.7.2 Typedef Documentation

2.7.2.1 CLPROD

```
typedef struct clProd* CLPROD
```

Estrutura que relaciona clientes com produtos.

2.7.2.2 CLSEARCHER

```
typedef struct clSearcher* CLSEARCHER
```

Estrutura que guarda clientes numa procura.

2.7.2.3 PRODCL

```
typedef struct prodCl* PRODCL
```

Estrutura que relaciona produtos com clientes.

2.7.2.4 PRSEARCHER

```
typedef struct prSearcher* PRSEARCHER
```

Estrutura que guarda produtos numa procura.

2.7.2.5 SALES

```
typedef struct sales* SALES
```

Estrutura das vendas.

2.7.3 Function Documentation

2.7.3.1 addBranch()

```
void addBranch (
    SALES sales,
    char * product,
    char * client,
    char type,
    float price,
    int units,
    int month )
```

Carrega uma venda a Estrutura de Vendas.

Parameters

<i>Estrutura</i>	de Vendas
<i>Cliente</i>	a adicionar
<i>Produto</i>	a adicionar
<i>Tipo</i>	da venda
<i>Custo</i>	unitario do produto
<i>Unidades</i>	vendidas
<i>Mes</i>	da venda

2.7.3.2 changeClientsProducts()

```
void changeClientsProducts (
    CLPROD cp,
    char * product,
    int month,
    int units,
    float price )
```

Altera a Estrutura que relaciona clientes com produtos, alterando a faturação do produto.

Parameters

<i>Estrutra</i>	a ser alterada
<i>Produto</i>	que sera alterado
<i>Mes</i>	da venda
<i>Unidades</i>	da vena
<i>Preço</i>	unitario da venda

2.7.3.3 changeProductsClients()

```
void changeProductsClients (
    PRODCL cl,
    char * client,
    char type )
```

Altera a Estrutura que relaciona produtos com clientes, alterando o tipo de venda que o cliente fez.

Parameters

<i>Estrutra</i>	a ser alterada
<i>Cliente</i>	que sera alterado
<i>Tipo</i>	de venda

2.7.3.4 changeTotalProducts()

```
void changeTotalProducts (
    double total[][2][12],
    char type,
    float price,
    int month )
```

Altera a Estrutura que relaciona produtos com clientes, aumentando o preço e o numero de vendas sofridas por um produto.

Parameters

<i>Estrutura</i>	a ser alterada
<i>Tipo</i>	da venda
<i>Preço</i>	unitario da venda
<i>Mes</i>	da venda

2.7.3.5 destroyClientsProducts()

```
void destroyClientsProducts (
    void * clPr )
```

Destruição da Estrutura que relaciona clientes com produtos.

Parameters

<i>Estrutura</i>	a ser destruida
------------------	-----------------

2.7.3.6 destroyCLS()

```
void destroyCLS (
    CLSEARCHER cls )
```

Função que elimina a estrutura de procura "CLSEARCHER".

Parameters

<i>Estrutura</i>	a ser destruida
------------------	-----------------

2.7.3.7 destroyProductsClients()

```
void destroyProductsClients (
    void * prCl )
```

Destruição da Estrutura que relaciona produtos com clientes.

Parameters

<i>Estrutura</i>	a ser destruida
------------------	-----------------

2.7.3.8 destroySales()

```
void destroySales (
    SALES sales )
```

Destruição da Estrutura das Vendas.

Parameters

<i>Estrutura</i>	a ser destruída
------------------	-----------------

2.7.3.9 findClientFavotiteProducts()

```
PRSEARCHER findClientFavotiteProducts (
    SALES * sales,
    char * client,
    int month,
    int * size )
```

Função que constroi a estrutura de procura "PRSEARCHER".

Parameters

<i>Array</i>	de Estrutura das vendas
<i>Cliente</i>	a ser procurado
<i>Mes</i>	a ser procurado
<i>Numero</i>	de produtos comprados por um cliente

Returns

Estrutura de procura "PRSEARCHER"

2.7.3.10 findCLS()

```
void findCLS (
    gpointer key,
    gpointer value,
    gpointer data )
```

Função auxiliar que encontra clientes.

Parameters

<i>Estrutura</i>	que relaciona produtos com clientes
<i>Flag</i>	para procurar clientes que fizeram compras de tipo N ou P

Returns

Numero de vendas

2.7.3.11 findData()

```
void findData (
    SALES * sales,
    int data[][3][2],
    char ** prods,
    int limit,
    int branch )
```

Parameters

<i>Array</i>	de estrutura das vendas
<i>Matriz</i>	que ir guardar os resultados dividido em filiais, vendas, e numero de clientes comprados
<i>Numero</i>	de produtos
<i>Filial</i>	a procurar

2.7.3.12 findProductSalesAndProfit()

```
void findProductSalesAndProfit (
    SALES * sales,
    char * productID,
    int month,
    double res[][4] )
```

Função que encontra as vendas e a faturação de um produto num certo mes.

Parameters

<i>Estrutura</i>	de filiais
<i>Produto</i>	a procura
<i>Mes</i>	em questao
<i>Array</i>	o qual sera guardado as respostas das vendas e faturação dividido em filiais

2.7.3.13 findProductSellesByClient()

```
void findProductSellesByClient (
    SALES sales,
    char * client,
    int * res )
```

Função que encontra as vendas feitas por um cliente.

Parameters

<i>Estrutura</i>	das vendas
<i>Cliente</i>	a procurar
<i>Array</i>	onde sera guardado as respostas

2.7.3.14 findSalesMerge()

```
char** findSalesMerge (
    GHashTable * merge,
    double qt[],
    int size,
    int limit,
    int flag )
```

Função encontra os 'n' produtos que mais faturou em todo ano (este executa a query 11 e query a partir de uma flag)

Parameters

<i>Hash</i>	table de todas as filiais juntas
<i>Tamanho</i>	da hash table
<i>Numero</i>	de produtos a ir buscar
<i>Flag</i>	para saber se estamos a executar a query 11 ou 12

Returns

Lista de produtos

2.7.3.15 getClientsAndUnitsCLS()

```
int getClientsAndUnitsCLS (
    CLSEARCHER cls,
    char *** clientes,
    int flag )
```

Função que retira os clientes e as unidades da Estrutura de procura "CLSEARCHER".

Parameters

<i>Estrutura</i>	de procura "CLSEARCHER"
<i>Lista</i>	de strings onde sera guardado os clientes
<i>Flag</i>	para determinar o tipo N ou P

Returns

Numero de unidades

2.7.3.16 getCLSearcher()

```
CLSEARCHER getCLSearcher (
    SALES sale,
    char * productID )
```

Função que encontra as clientes que compraram um produto.

Parameters

<i>Estrutura</i>	das vendas
<i>Produto</i>	a procurar

Returns

Estrutura que guarda os clientes dividindo-os em compras N e P

2.7.3.17 getProductsAndUnitsPRS()

```
char** getProductsAndUnitsPRS (
    PRSEARCHER prs,
    int size,
    int qt[] )
```

Função encontra os produtos da estrutura "PRSEARCHER" e coloca-os numa lista de strings.

Parameters

<i>Estrutura</i>	de procura "PRSEARCHER"
<i>Tamanho</i>	da lista

Returns

Lista de produtos

2.7.3.18 initClientProducts()

```
CLPROD initClientProducts ( )
```

Inicialização da Estrutura que relaciona Clientes com Produtos.

Returns

Estrutura que relaciona Clientes com Produtos inicializada

2.7.3.19 initProductsClients()

```
PRODCL initProductsClients ( )
```

Inicialização da Estrutura que relaciona Produtos com Clientes.

Returns

Estrutura que relaciona Produtos com Clientes inicializada

2.7.3.20 initSales()

```
SALES initSales ( )
```

Inicializador da Estrutura das Vendas.

Returns

Estrutura das Vendas

2.7.3.21 mergeBranches()

```
GHashTable* mergeBranches (
    SALES * sales )
```

Função que junta as filiais numa unica hash table.

Parameters

Array	de Estruturas das vendas
-------	--------------------------

Returns

Hash table retornada

2.7.3.22 mergeBranchesAndMonths()

```
GHashTable* mergeBranchesAndMonths (
    SALES * sales,
    char * client )
```

Função que junta as filiais devididas em meses numa unica hash table.

Parameters

<i>Array</i>	de Estruturas das vendas
--------------	--------------------------

Returns

Hash table retornada

2.7.3.23 numberOfSales()

```
int numberOfSales (
    PRODCL pc,
    int flag )
```

Função que encontra o numero de vendas feitas.

Parameters

<i>Estrutura</i>	que relaciona produtos com clientes
<i>Flag</i>	para procurar clientes que fizeram compras de tipo N ou P

Returns

Numero de vendas

2.7.3.24 sizeOfProducts()

```
int sizeOfProducts (
    SALES sales,
    char * client,
    int month )
```

Função que calcula o numero de produtos comprados por um cliente.

Parameters

<i>Estrutura</i>	das vendas
<i>Cliente</i>	a ser procurado
<i>Mes</i>	a ser procurado

Returns

numero de produtos comprados

2.7.3.25 sort()

```
int sort (
    const void * one,
    const void * two )
```

Função auxiliar que retorna se o segundo parametro é maior do que o primeiro.

Parameters

<i>String</i>	1
<i>String</i>	2

Returns

Resposta da comparação de duas strings

2.8 sgv.h File Reference

Modulo de tratamento da Base de Dados.

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include "clients.h"
#include "products.h"
#include "sales.h"
```

Typedefs

- typedef struct sgv * [SGV](#)
Base de Dados.

Functions

- [SGV initialize_SGV](#) ()
Inicializador da Base de Dados.
- void [destructionSGV](#) ([SGV](#) sgv)
Destruição da Base de Dados.
- [SGV loadSGV](#) ([SGV](#) sgv, char *clientsFilePath, char *productsFilePath, char *salesFilePath)

- Carrega Bases de Dados (SGV) com os ficheiros.*

 - void [readClients](#) (SGV sgv, char *clientsFilePath)

Carrega Estrutura dos clientes a partir do ficheiro dos clientes.

 - void [readProducts](#) (SGV sgv, char *productsFilePath)

Carrega Estrutura dos produtos a partir do ficheiro dos produtos.

 - void [readSales](#) (SGV sgv, char *salesFilePath)

Carrega Estrutura das vendas a partir do ficheiro das vendas.

 - void [changeTotalMonth](#) (SGV sgv, float price, int units, int month)

Altera o numero de vendas faturação total por meses.

 - int [validadeSale](#) (SGV sgv, char **token)

Verifica se uma venda é valida.

 - int [validateProduct](#) (char *buffer)

Verifica se um produto é bem contruido.

 - int [validatePrice](#) (char *price)

Verifica se um preço é bem contruido.

 - int [validateUnits](#) (char *units)

Verifica se as unidades sao bem contruidas.

 - int [validateType](#) (char *type)

Verifica se o tipo de venda é bem contruida.

 - int [validateClient](#) (char *buffer)

Verifica se um cliente é bem contruido.

 - int [validateMonth](#) (char *month)

Verifica se o mes da venda é bem contruida.

 - int [validateBranch](#) (char *branch)

Verifica se a filial da venda é bem contruida.

 - [PRODUCTS](#) [getArrayProductsByLetter](#) (SGV sgv, char letter)

Função que encontra o array de produtos correspondente a uma letra.

 - [PRODUCTS](#) * [getArrayProducts](#) (SGV sgv)

Função que encontra o array de produtos.

 - [SALES](#) * [getArraySales](#) (SGV sgv)

Função que retorna o array de filias.

 - [SALES](#) [getSales](#) (SGV sgv, int branch)

Função que retorna um Estrutura de Filias.

 - [CLIENTS](#) [getClients](#) (SGV sgv)

Função que retorna a Estrutura dos Clientes.

 - int * [getLinesValidatedAndRead](#) (SGV sgv)

Função que retorna as linhas lidas e as validas da leitura dos files.

 - void [findSalesAndProfit](#) (SGV sgv, int minMonth, int maxMonth, double res[])

Função que retorna a Estrutura dos Clientes.

2.8.1 Detailed Description

Modulo de tratamento da Base de Dados.

2.8.2 Typedef Documentation

2.8.2.1 SGV

```
typedef struct sgv* SGV
```

Base de Dados.

2.8.3 Function Documentation

2.8.3.1 changeTotalMonth()

```
void changeTotalMonth (
    SGV sgv,
    float price,
    int units,
    int month )
```

Altera o numero de vendas faturação total por meses.

Parameters

<i>Base</i>	de Dados (SGV)
<i>Preço</i>	unitario da vendas
<i>Unidades</i>	da venda
<i>Mes</i>	da venda

2.8.3.2 destructionSGV()

```
void destructionSGV (
    SGV sgv )
```

Destruição da Base de Dados.

Parameters

<i>Estrutura</i>	a ser destruida
------------------	-----------------

2.8.3.3 findSalesAndProfit()

```
void findSalesAndProfit (
    SGV sgv,
```

```
int minMonth,  
int maxMonth,  
double res[] )
```

Função que retorna a Estrutura dos Clientes.

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Mes</i>	minimo
<i>Mes</i>	maximo
<i>Array</i>	que ira guardar a resposta

2.8.3.4 `getArrayProducts()`

```
PRODUCTS* getArrayProducts (  
    SGV sgv )
```

Função que encontra o array de produtos.

Returns

Array da Estrutura dos produtos

2.8.3.5 `getArrayProductsByLetter()`

```
PRODUCTS getArrayProductsByLetter (  
    SGV sgv,  
    char letter )
```

Função que encontra o array de produtos correspondente a uma letra.

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Letra</i>	a qual ir corresponde o array

Returns

Estrutura dos produtos

2.8.3.6 `getArraySales()`

```
SALES* getArraySales (
    SGV sgV )
```

Função que retorna o array de filias.

Parameters

<i>Estrutura</i>	de Dados (SGV)
------------------	----------------

Returns

Estrutura de filias

2.8.3.7 `getClients()`

```
CLIENTS getClients (
    SGV sgV )
```

Função que retorna a Estrutura dos Clientes.

Parameters

<i>Estrutura</i>	de Dados (SGV)
------------------	----------------

Returns

Estrutura dos Clientes

2.8.3.8 `getLinesValidatedAndRead()`

```
int* getLinesValidatedAndRead (
    SGV sgV )
```

Função que retorna as linhas lidas e as validas da leitura dos files.

Parameters

<i>Estrutura</i>	de Dados (SGV)
------------------	----------------

Returns

Array com o numero das linhas lidas e as validas da leitura dos files

2.8.3.9 getSales()

```
SALES getSales (
    SGV sgv,
    int branch )
```

Função que retorna um Estrutura de Filias.

Parameters

<i>Estrutura</i>	de Dados (SGV)
<i>Filia</i>	a ser procurada

Returns

Estrutura de filias

2.8.3.10 initialize_SGV()

```
SGV initialize_SGV ( )
```

Inicializador da Base de Dados.

Returns

Estrutura SGV

2.8.3.11 loadSGV()

```
SGV loadSGV (
    SGV sgv,
    char * clientsFilePath,
    char * productsFilePath,
    char * salesFilePath )
```

Carrega Bases de Dados (SGV) com os ficheiros.

Parameters

<i>Ficheiro</i>	dos clientes
<i>Ficheiro</i>	dos produtos
<i>Ficheiro</i>	das vendas

Returns

Bases de Dados (SGV) carregado

2.8.3.12 readClients()

```
void readClients (
    SGV sgv,
    char * clientsFilePath )
```

Carrega Estrutura dos clientes a partir do ficheiro dos clientes.

Parameters

<i>Base</i>	de Dados (SGV)
<i>Ficheiro</i>	dos clientes

2.8.3.13 readProducts()

```
void readProducts (
    SGV sgv,
    char * productsFilePath )
```

Carrega Estrutura dos produtos a partir do ficheiro dos produtos.

Parameters

<i>Base</i>	de Dados (SGV)
<i>Ficheiro</i>	dos produtos

2.8.3.14 readSales()

```
void readSales (
    SGV sgv,
    char * salesFilePath )
```

Carrega Estrutura das vendas a partir do ficheiro das vendas.

Parameters

<i>Base</i>	de Dados (SGV)
<i>Ficheiro</i>	das vendas

2.8.3.15 validadeSale()

```
int validadeSale (
    SGV sgv,
    char ** token )
```

Verifica se uma venda é valida.

Parameters

<i>Base</i>	de Dados (SGV)
<i>Apontador</i>	de strings a serem validadas

2.8.3.16 validateBranch()

```
int validateBranch (
    char * branch )
```

Verifica se a filial da venda é bem contruida.

Parameters

<i>String</i>	a ser validada
---------------	----------------

2.8.3.17 validateClient()

```
int validateClient (
    char * buffer )
```

Verifica se um cliente é bem contruido.

Parameters

<i>String</i>	a ser validada
---------------	----------------

2.8.3.18 validateMonth()

```
int validateMonth (
    char * month )
```

Verifica se o mes da venda é bem contruida.

Parameters

<i>String</i>	a ser validada
---------------	----------------

2.8.3.19 validatePrice()

```
int validatePrice (
    char * price )
```

Verifica se um preço é bem contruido.

Parameters

<i>String</i>	a ser validada
---------------	----------------

2.8.3.20 validateProduct()

```
int validateProduct (
    char * buffer )
```

Verifica se um produto é bem contruido.

Parameters

<i>String</i>	a ser validada
---------------	----------------

2.8.3.21 validateType()

```
int validateType (
    char * type )
```

Verifica se o tipo de venda é bem contruida.

Parameters

<i>String</i>	a ser validada
---------------	----------------

2.8.3.22 validateUnits()

```
int validateUnits (
    char * units )
```

Verifica se as unidades sao bem contruidas.

Parameters

<i>String</i>	a ser validada
---------------	----------------

Index

- addBranch
 - sales.h, [43](#)
- addClient
 - clients.h, [4](#)
- addProduct
 - products.h, [33](#)
- changeBranchClients
 - clients.h, [4](#)
- changeBranchProducts
 - products.h, [33](#)
- changeClientsProducts
 - sales.h, [44](#)
- changeProductsClients
 - sales.h, [44](#)
- changeTotalMonth
 - sgv.h, [55](#)
- changeTotalProducts
 - sales.h, [44](#)
- clienteMesPalavras
 - menu.h, [20](#)
- clientePalavra
 - menu.h, [21](#)
- clientesQuery9
 - menu.h, [21](#)
- CLIENTS
 - clients.h, [4](#)
- clients.h, [3](#)
 - addClient, [4](#)
 - changeBranchClients, [4](#)
 - CLIENTS, [4](#)
 - destroyClients, [5](#)
 - findClientesOfAllBranches, [5](#)
 - findClientsNeverBoughtCount, [5](#)
 - initClients, [5](#)
 - initCLS, [6](#)
 - listOfClients, [6](#)
 - numberOfClientsNeverBought, [6](#)
 - numberOfClientsOfAllBranches, [7](#)
 - sortByLetter, [7](#)
- CLPROD
 - sales.h, [42](#)
- CLSEARCHER
 - sales.h, [42](#)
- dadosClientesMes
 - menu.h, [21](#)
- dadosCompras
 - menu.h, [21](#)
- dadosComprasG
 - menu.h, [21](#)
- dadosQuery10
 - menu.h, [21](#)
- dadosQuery11
 - menu.h, [22](#)
- dadosQuery12
 - menu.h, [22](#)
- dadosTotalVendas
 - menu.h, [22](#)
- dataPalavra
 - menu.h, [22](#)
- destructionSGV
 - sgv.h, [55](#)
- destroyClients
 - clients.h, [5](#)
- destroyClientsProducts
 - sales.h, [45](#)
- destroyCLS
 - sales.h, [45](#)
- destroyProducts
 - products.h, [34](#)
- destroyProductsClients
 - sales.h, [45](#)
- destroyPS
 - products.h, [34](#)
- destroySales
 - sales.h, [45](#)
- destroySGV
 - interface.h, [13](#)
- enterToContinue
 - menu.h, [22](#)
- fatUniPalavras
 - menu.h, [22](#)
- FatUniPalavrasG
 - menu.h, [23](#)
- files.h, [8](#)
 - getFiles, [8](#)
 - verifyFiles, [8](#)
- filesInfo
 - menu.h, [23](#)
- filiaisPalavras
 - menu.h, [23](#)
- findClientesOfAllBranches
 - clients.h, [5](#)
- findClientFavotiteProducts
 - sales.h, [46](#)
- findClientsNeverBoughtCount
 - clients.h, [5](#)

- findCLS
 - sales.h, [46](#)
- findData
 - sales.h, [47](#)
- findProductSalesAndProfit
 - sales.h, [47](#)
- findProductsByLetter
 - products.h, [34](#)
- findProductSellesByClient
 - sales.h, [47](#)
- findProductsNeverBought
 - products.h, [35](#)
- findProductsNeverBoughtCount
 - products.h, [35](#)
- findSalesAndProfit
 - sgv.h, [55](#)
- findSalesMerge
 - sales.h, [49](#)
- getArrayProducts
 - sgv.h, [56](#)
- getArrayProductsByLetter
 - sgv.h, [56](#)
- getArraySales
 - sgv.h, [56](#)
- getBranch
 - gets.h, [10](#)
- getBranches2
 - menu.h, [23](#)
- getClient
 - gets.h, [10](#)
- getClient2
 - menu.h, [23](#)
- getClientAndMonth2
 - menu.h, [23](#)
- getClientAndNumber
 - menu.h, [23](#)
- getClientFavoriteProducts
 - interface.h, [13](#)
- getClients
 - sgv.h, [57](#)
- getClientsAndProductsNeverBoughtCount
 - interface.h, [13](#)
- getClientsAndUnitsCLS
 - sales.h, [49](#)
- getClientsOfAllBranches
 - interface.h, [14](#)
- getClientTopProfitProducts
 - interface.h, [14](#)
- getCLSearcher
 - sales.h, [50](#)
- getCurrentFilesInfo
 - interface.h, [14](#)
- getEnter
 - menu.h, [24](#)
- getFiles
 - files.h, [8](#)
- getLetter
 - gets.h, [10](#)
- getLetter2
 - menu.h, [24](#)
- getLinesValidatedAndRead
 - sgv.h, [57](#)
- getMaxMonth
 - gets.h, [10](#)
- getMinAndMaxMonth2
 - menu.h, [24](#)
- getMonth
 - gets.h, [10](#)
- getNumber
 - gets.h, [11](#)
- getNumber2
 - menu.h, [24](#)
- getOneBranch
 - gets.h, [11](#)
- getProduct
 - gets.h, [11](#)
- getProductAndBranch2
 - menu.h, [24](#)
- getProductAndMonth2
 - menu.h, [24](#)
- getProductSalesAndProfit
 - interface.h, [14](#)
- getProductsAndUnitsPRS
 - sales.h, [50](#)
- getProductsBoughtByClient
 - interface.h, [16](#)
- getProductsBuyers
 - interface.h, [16](#)
- getProductsNeverBought
 - interface.h, [16](#)
- getProductsOfPS
 - products.h, [35](#)
- getProductsStartedByLetter
 - interface.h, [17](#)
- gets.h, [9](#)
 - getBranch, [10](#)
 - getClient, [10](#)
 - getLetter, [10](#)
 - getMaxMonth, [10](#)
 - getMonth, [10](#)
 - getNumber, [11](#)
 - getOneBranch, [11](#)
 - getProduct, [11](#)
- getSales
 - sgv.h, [58](#)
- getSalesAndProfit
 - interface.h, [17](#)
- getTopSelledProducts
 - interface.h, [17](#)
- getTotalOfPS
 - products.h, [36](#)
- infoPalavra
 - menu.h, [24](#)
- initClientProducts
 - sales.h, [50](#)
- initClients

- clients.h, 5
- initCLS
 - clients.h, 6
- initialize_SGV
 - sgv.h, 58
- initProducts
 - products.h, 36
- initProductsClients
 - sales.h, 51
- initProductSearch
 - products.h, 36
- initSales
 - sales.h, 51
- initSGV
 - interface.h, 18
- interface.h, 12
 - destroySGV, 13
 - getClientFavoriteProducts, 13
 - getClientsAndProductsNeverBoughtCount, 13
 - getClientsOfAllBranches, 14
 - getClientTopProfitProducts, 14
 - getCurrentFilesInfo, 14
 - getProductSalesAndProfit, 14
 - getProductsBoughtByClient, 16
 - getProductsBuyers, 16
 - getProductsNeverBought, 16
 - getProductsStartedByLetter, 17
 - getSalesAndProfit, 17
 - getTopSelledProducts, 17
 - initSGV, 18
 - loadSGVFromFiles, 18
 - showFilesInfo, 18
- iter_all
 - products.h, 37
- letrasSGV
 - menu.h, 25
- listOfClients
 - clients.h, 6
- listOfProducts
 - products.h, 37
- loading
 - menu.h, 25
- loadSGV
 - sgv.h, 58
- loadSGVFromFiles
 - interface.h, 18
- menu.h, 19
 - clienteMesPalavras, 20
 - clientePalavra, 21
 - clientesQuery9, 21
 - dadosClientesMes, 21
 - dadosCompras, 21
 - dadosComprasG, 21
 - dadosQuery10, 21
 - dadosQuery11, 22
 - dadosQuery12, 22
 - dadosTotalVendas, 22
- dataPalavra, 22
- enterToContinue, 22
- fatUniPalavras, 22
- FatUniPalavrasG, 23
- filesInfo, 23
- filiaisPalavras, 23
- getBranches2, 23
- getClient2, 23
- getClientAndMonth2, 23
- getClientAndNumber, 23
- getEnter, 24
- getLetter2, 24
- getMinAndMaxMonth2, 24
- getNumber2, 24
- getProductAndBranch2, 24
- getProductAndMonth2, 24
- infoPalavra, 24
- letrasSGV, 25
- loading, 25
- menuPalavra, 25
- miniMenuQuery2, 25
- miniMenuQuery3, 25
- miniMenuQuery5, 25
- miniMenuQuery6, 25
- miniMenuQuery9, 26
- mostraClientesPalavras, 26
- mostraClientesQuery5, 26
- mostraContador, 26
- mostraFilial, 26
- mostraLetra, 26
- mostraProdutosQuery2, 27
- parteDeBaixoMenu, 27
- parteDeBaixoTabela, 27
- parteDeBaixoTempos, 27
- parteDeCimaMenu, 27
- parteDeCimaTabela, 27
- parteDeCimaTempos, 27
- printaDadosTotalVendas, 28
- produtoFilialPalavras, 28
- produtoMesPalavras, 28
- query11Palavras, 28
- setasPalavras, 28
- showAndGetOptionsMenu2, 28
- showClientFavoriteProducts, 28
- showClientsAndProductsNeverBoughtCount, 29
- showClientsOfAllBranches, 29
- showClientTopProfitProducts, 29
- showDestructionTime, 29
- showMenu1, 29
- showMenu2, 30
- showMenuOption, 30
- showProductSalesAndProfit, 30
- showProductsBoughtByClient, 30
- showProductsBuyers, 30
- showProductsNeverBought, 30
- showProductsStartbyLetter, 31
- showSalesAndProfit, 31
- showTopSoldProducts, 31

- visaoGlobalPalvra, 31
- menuPalavra
 - menu.h, 25
- mergeBranches
 - sales.h, 51
- mergeBranchesAndMonths
 - sales.h, 51
- miniMenuQuery2
 - menu.h, 25
- miniMenuQuery3
 - menu.h, 25
- miniMenuQuery5
 - menu.h, 25
- miniMenuQuery6
 - menu.h, 25
- miniMenuQuery9
 - menu.h, 26
- mostraClientesPalavras
 - menu.h, 26
- mostraClientesQuery5
 - menu.h, 26
- mostraContador
 - menu.h, 26
- mostraFilial
 - menu.h, 26
- mostraLetra
 - menu.h, 26
- mostraProdutosQuery2
 - menu.h, 27
- numberOfClientsNeverBought
 - clients.h, 6
- numberOfClientsOfAllBranches
 - clients.h, 7
- numberOfProductsNeverBought
 - products.h, 37
- numberOfSales
 - sales.h, 52
- parteDeBaixoMenu
 - menu.h, 27
- parteDeBaixoTabela
 - menu.h, 27
- parteDeBaixoTempos
 - menu.h, 27
- parteDeCimaMenu
 - menu.h, 27
- parteDeCimaTabela
 - menu.h, 27
- parteDeCimaTempos
 - menu.h, 27
- printaDadosTotalVendas
 - menu.h, 28
- PRODCL
 - sales.h, 42
- PRODUCTS
 - products.h, 33
- products.h, 31
 - addProduct, 33
 - changeBranchProducts, 33
 - destroyProducts, 34
 - destroyPS, 34
 - findProductsByLetter, 34
 - findProductsNeverBought, 35
 - findProductsNeverBoughtCount, 35
 - getProductsOfPS, 35
 - getTotalOfPS, 36
 - initProducts, 36
 - initProductSearch, 36
 - iter_all, 37
 - listOfProducts, 37
 - numberOfProductsNeverBought, 37
 - PRODUCTS, 33
 - PRODUCTSEARCH, 33
 - productsNeverBought, 38
 - productsNeverBought0, 38
 - productsNeverBought1, 39
 - productsNeverBought2, 39
 - sizeOfProductsNeverBoughtAllBranches, 40
 - sizeOfProductsNeverBoughtEveryBranch, 40
- PRODUCTSEARCH
 - products.h, 33
- productsNeverBought
 - products.h, 38
- productsNeverBought0
 - products.h, 38
- productsNeverBought1
 - products.h, 39
- productsNeverBought2
 - products.h, 39
- produtoFilialPalavras
 - menu.h, 28
- produtoMesPalavras
 - menu.h, 28
- PRSEARCHER
 - sales.h, 43
- query11Palavras
 - menu.h, 28
- readClients
 - sgv.h, 59
- readProducts
 - sgv.h, 59
- readSales
 - sgv.h, 59
- SALES
 - sales.h, 43
- sales.h, 40
 - addBranch, 43
 - changeClientsProducts, 44
 - changeProductsClients, 44
 - changeTotalProducts, 44
 - CLPROD, 42
 - CLSEARCHER, 42
 - destroyClientsProducts, 45
 - destroyCLS, 45

- destroyProductsClients, [45](#)
- destroySales, [45](#)
- findClientFavotiteProducts, [46](#)
- findCLS, [46](#)
- findData, [47](#)
- findProductSalesAndProfit, [47](#)
- findProductSellesByClient, [47](#)
- findSalesMerge, [49](#)
- getClientsAndUnitsCLS, [49](#)
- getCLSearcher, [50](#)
- getProductsAndUnitsPRS, [50](#)
- initClientProducts, [50](#)
- initProductsClients, [51](#)
- initSales, [51](#)
- mergeBranches, [51](#)
- mergeBranchesAndMonths, [51](#)
- numberOfSales, [52](#)
- PRODCL, [42](#)
- PRSEARCHER, [43](#)
- SALES, [43](#)
- sizeOfProducts, [52](#)
- sort, [53](#)
- setasPalavras
 - menu.h, [28](#)
- SGV
 - sgv.h, [54](#)
- sgv.h, [53](#)
 - changeTotalMonth, [55](#)
 - destructionSGV, [55](#)
 - findSalesAndProfit, [55](#)
 - getArrayProducts, [56](#)
 - getArrayProductsByLetter, [56](#)
 - getArraySales, [56](#)
 - getClients, [57](#)
 - getLinesValidatedAndRead, [57](#)
 - getSales, [58](#)
 - initialize_SGV, [58](#)
 - loadSGV, [58](#)
 - readClients, [59](#)
 - readProducts, [59](#)
 - readSales, [59](#)
 - SGV, [54](#)
 - validadeSale, [60](#)
 - validateBranch, [60](#)
 - validateClient, [60](#)
 - validateMonth, [60](#)
 - validatePrice, [62](#)
 - validateProduct, [62](#)
 - validateType, [62](#)
 - validateUnits, [62](#)
- showAndGetOptionMenu2
 - menu.h, [28](#)
- showClientFavoriteProducts
 - menu.h, [28](#)
- showClientsAndProductsNeverBoughtCount
 - menu.h, [29](#)
- showClientsOfAllBranches
 - menu.h, [29](#)
- showClientTopProfitProducts
 - menu.h, [29](#)
- showDestructionTime
 - menu.h, [29](#)
- showFilesInfo
 - interface.h, [18](#)
- showMenu1
 - menu.h, [29](#)
- showMenu2
 - menu.h, [30](#)
- showMenuOption
 - menu.h, [30](#)
- showProductSalesAndProfit
 - menu.h, [30](#)
- showProductsBoughtByClient
 - menu.h, [30](#)
- showProductsBuyers
 - menu.h, [30](#)
- showProductsNeverBought
 - menu.h, [30](#)
- showProductsStartbyLetter
 - menu.h, [31](#)
- showSalesAndProfit
 - menu.h, [31](#)
- showTopSoldProducts
 - menu.h, [31](#)
- sizeOfProducts
 - sales.h, [52](#)
- sizeOfProductsNeverBoughtAllBranches
 - products.h, [40](#)
- sizeOfProductsNeverBoughtEveryBranch
 - products.h, [40](#)
- sort
 - sales.h, [53](#)
- sortByLetter
 - clients.h, [7](#)
- validadeSale
 - sgv.h, [60](#)
- validateBranch
 - sgv.h, [60](#)
- validateClient
 - sgv.h, [60](#)
- validateMonth
 - sgv.h, [60](#)
- validatePrice
 - sgv.h, [62](#)
- validateProduct
 - sgv.h, [62](#)
- validateType
 - sgv.h, [62](#)
- validateUnits
 - sgv.h, [62](#)
- verifyFiles
 - files.h, [8](#)
- visaoGlobalPalvra
 - menu.h, [31](#)