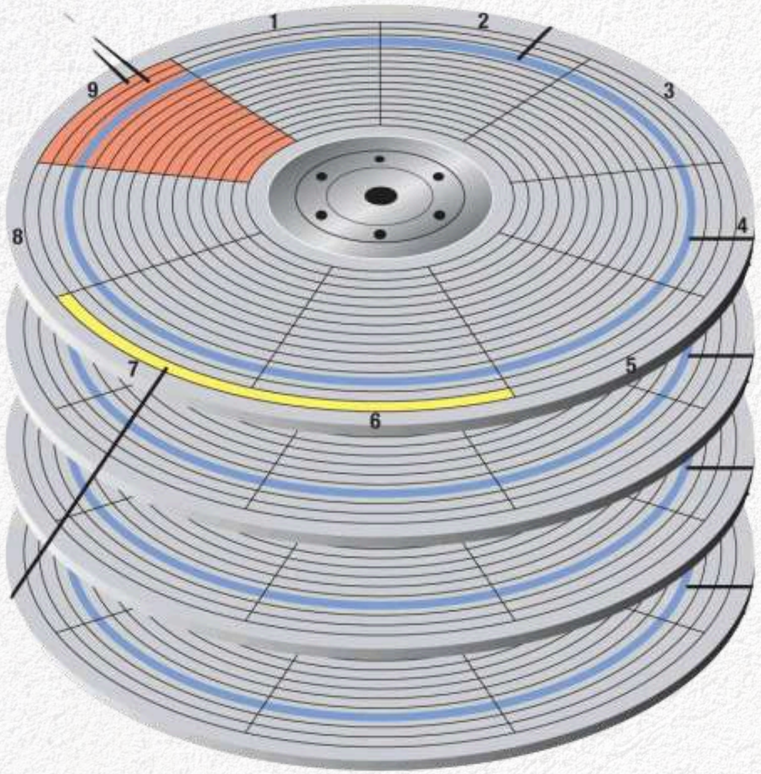
MEGATRON

By Misael Palomino

Disco



Disco Ideal características:

- Tam_sector: 1024 bytes
- Cantidad de Platos: 6
- Superficies : 2
- Pistas x Superficie: 4
- Sectores x pista: 8
- Sectores x Bloque: 2

Capacidad total: 393,216 bytes

¿Porque?

- Tam_metadata: 51 bytes por registro
- Debido a la forma de construcción de bloques donde incrementa
- Superficie->Plato->Sector->Pista

Capacidad

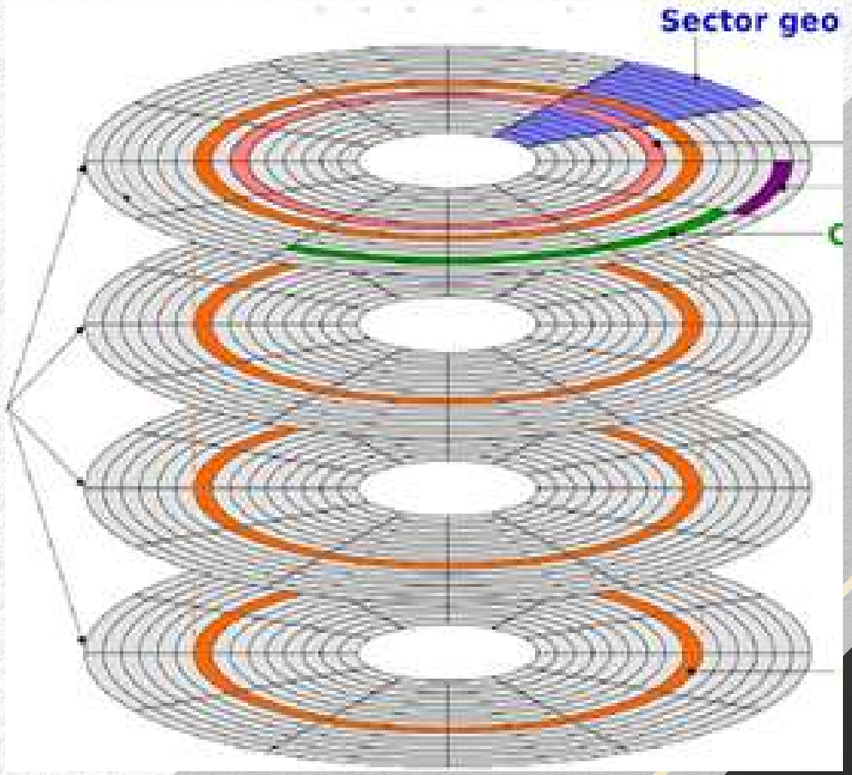
- Titanic 891 registros
- tam_máximo: 206
- Total: 183546.
- 223 sectores

- Housing 546 registros
- tam_máximo: 114
- Total: 62700 bytes.
- 62 sectores

```
Metadata_______
0_______0_______0_______0_______
1_______1_______0_______0_______
```

¿Cuál es el orden de insercion?

```
void avanzarPosicion(int &plato, int &superficie, int &sector, int &pista, Disco& disco) {
    superficie++;
    if (superficie >= disco.getNumSuperficies()) {
        superficie = 0;
        plato++;
        if (plato >= disco.getNumPlatos()) {
            plato = 0;
            sector++;
            if (sector >= disco.getNumSectores()) {
                sector = 0;
                pista++;
                if (pista >= disco.getNumPistas()) {
                    throw runtime_error("Se superó la cantidad total de posiciones.");
                }
            }
        }
    }
}
```



```cpp
void mostrar_arbol() {
    cout << "\n===== ARBOL DE CREACION DEL DISCO =====\n";
    for (int i = 0; i < numPlatos; ++i) {
        cout << "Plato " << i << ":\n";
        for (int j = 0; j < numSuperficies; ++j) {
            cout << "   Superficie " << j << ":\n";
            for (int k = 0; k < numPistas; ++k) {
                cout << "     Pista " << k << ":\n";
                for (int l = 0; l < numSectores; ++l) {
                    cout << "       Sector " << l << '\n';
                }
            }
        }
    }
}
```

```
===== ARBOL DE CREACION DEL DISCO =====
Plato 0:
   Superficie 0:
     Pista 0:
       Sector 0
       Sector 1
       Sector 2
       Sector 3
       Sector 4
       Sector 5
       Sector 6
       Sector 7
     Pista 1:
       Sector 0
       Sector 1
       Sector 2
       Sector 3
       Sector 4
       Sector 5
       Sector 6
       Sector 7
     Pista 2:
       Sector 0
       Sector 1
       Sector 2
       Sector 3
       Sector 4
       Sector 5
       Sector 6
       Sector 7
     Pista 3:
       Sector 0
```

```
Disco(int numPlatos, int numSuperficies, int numPistas, int numSectores, size_t tam, int SectPorBloq
    : numPlatos(numPlatos), numSuperficies(numSuperficies),
      numPistas(numPistas), numSectores(numSectores), tam_sector(tam) , sectoresPorBloque(SectPorBloc
    string rutaBase = "Disco";
    for (int i = 0; i < numPlatos; ++i) {
        platos.emplace_back(i, numSuperficies, numPistas, numSectores, rutaBase, tam);
    }
    int cantidad_bloques_inc=numPlatos*numSuperficies*numPistas*numSectores;
    cantidad_bloques=cantidad_bloques_inc/sectoresPorBloque;
    if(cantidad_bloques_inc%sectoresPorBloque>0){
        cantidad_bloques++;
    }
}
```

```
===== CREACION DEL DISCO =====
Numero de platos: 6
Numero de pistas por superficie: 4
Numero de sectores por pista: 8
Tamano del sector (Bytes): 1024
Numero de sectores por bloque: 2
 Disco creado con ├®xito.
Capacidad en bytes: 393216
Capacidad en MB: 0.375
Capacidad_disponible: 393216
Cantidad total de bloques: 192
Capacidad de Bloque: 2048
Cantidad de Bloques por pista: 4
Cantidad de Bloques por plato: 32
```

```
class Superficie {
public:
    Superficie(int id, int numPistas, int numSectores, const string& rutaBase, size_t tam) {
        string rutaSuperficie = rutaBase + "\\Superficie_" + to_string(id);
        for (int i = 0; i < numPistas; ++i) {
            pistas.emplace_back(i, rutaSuperficie, numSectores, tam);
        }
    }
```

```
class Sector {
private:
    string path_archivo;
    size_t tam_sector;

public:
    Sector(const string& path, int tam) : path_archivo(path), tam_sector(tam) {
        ofstream archivo(path_archivo, ios::app);
        archivo.seekp(0, ios::end);
        archivo.close();
    }
```

Longitud Fija

insertar 1 registro teclado

Disco > Plato_0 > Superficie_1 > Pista_0 > Sector_1.txt

```
1   _________1532
2   _________-1
3
```

```
Seleccione una opcion: 16
Nombre Archivo: Titanic
=== Ingrese los datos del nuevo registro para la relaci├│n 'Titanic' ===
PassengerId (int, max 10 chars): 1
Survived (int, max 10 chars): 0
Pclass (int, max 10 chars): 3
Name (str, max 82 chars): Braund, Mr. Owen Harris
Sex (str, max 6 chars): male
Age (int, max 10 chars): 22
SibSp (int, max 10 chars): 1
Parch (int, max 10 chars): 0
Ticket (str, max 18 chars): A/5
Fare (float, max 8 chars): 7.25
Cabin (str, max 15 chars): C85
Embarked (str, max 5 chars): S
[INFO] Registro insertado en sector: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
 Registro insertado en bloque existente.
 Registro procesado y almacenado correctamente.
Capacidad total en bytes: 65536
Capacidad Disponible: 65067
Capacidad Ocupada: 469
```

```
________1326
_________-1
_________1#_________0#_________3#Braund, Mr. Owen Harris_____________________________________________#male__#________22#_________1#_________0
#A/5_____________#____7.25#C85___________#S____
```

Longitud Fija

insertar N registros

```
16. Ingresar 1 Registro L.Fija
Seleccione una opcion: 4
Nombre Archivo: Titanic
Cantidad Registros: 4
Tamaño de registro: 206
Usando bloque existente con espacio: 1326
[INFO] Registro insertado en sector: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
Carga finalizada. Registros insertados: 4 / 4. Bloques utilizados: 1
Capacidad total en bytes: 65536
Capacidad Disponible: 64033
Capacidad Ocupada: 1503
```

Bloque > Titanic0.txt

```
1  _________502
2  ________-1
3  _________1#________0#________3#Braund, Mr. Owen Harris_____________________________
4  _________1#________0#________3#Braund, Mr. Owen Harris_____________________________
5  _________2#________1#________1#Cumings, Mrs. John Bradley (Florence Briggs Thayer)_
6  _________3#________1#________3#Heikkinen, Miss. Laina_____________________________
7  _________4#________1#________1#Futrelle, Mrs. Jacques Heath (Lily May Peel)_______
8
```

Longitud Fija

insertar todo CSV

```
2. Select From * L.Fija
3. Cargar todo CSV L.fija
4. Agregar N registros L.fija
5. Salir
6. Select From Where
7. Eliminar un registro
8. Mostrar Bloques Libres
9. Mostrar Bloques Ocupados
10. Cargar todo CSV L.Variable
11. Agregar N registros L.Variable
12. SELECT FROM* L.Variable
13. Eliminar un registo L.Variable
14. Mostrar Sectores Ocupados
15. Ingresar 1 Registro L.Variable
16. Ingresar 1 Registro L.Fija
Seleccione una opcion: 3
Nombre Archivo: Titanic
```

```
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_0\Pista_2\Sector_3.txt
Nuevo bloque asignado para la relacion
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_0\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_0\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_0\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_1\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_1\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_1\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_1\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_3\Superficie_0\Pista_2\Sector_3.txt
Nuevo bloque asignado para la relacion
[INFO] Registro insertado en sector: Disco\Plato_3\Superficie_0\Pista_2\Sector_3.txt
[INFO] Registro insertado en sector: Disco\Plato_3\Superficie_0\Pista_2\Sector_3.txt
Carga finalizada. Bloques utilizados: 112
Capacidad total en bytes: 393216
Capacidad Disponible: 165011
Capacidad Ocupada: 228205
```

```
_______1426
_______-1
______889#_______0#_______3#Johnston, Miss. Catherine Helen Carrie_____________________________
______890#_______1#_______1#Behr, Mr. Karl Howell_____________________________________________
______891#_______0#_______3#Dooley, Mr. Patrick_______________________________________________
```

Longitud Fija

Sector sin suficiente espacio/LLeno

```
Numero de platos: 4
Numero de pistas por superficie: 4
Numero de sectores por pista: 4
Tamano del sector (Bytes): 512
Numero de sectores por bloque: 3
 Disco creado con ├®xito.
Capacidad en bytes: 65536
Capacidad en MB: 0.0625
Capacidad_disponible: 65536
Cantidad total de bloques: 43
Capacidad de Bloque: 1536
Cantidad de Bloques por pista: 1
Cantidad de Bloques por plato: 10
```

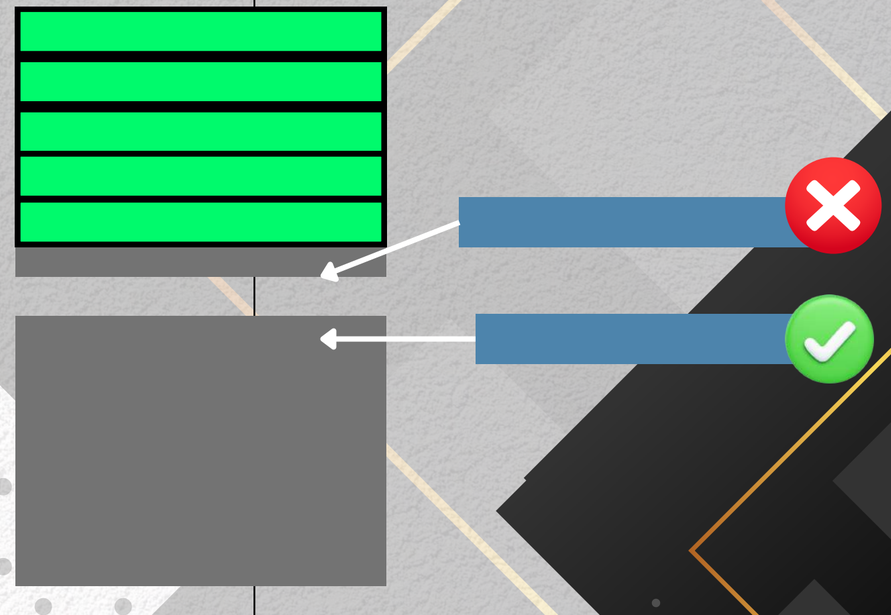
```
Seleccione una opcion: 4
Nombre Archivo: Titanic
Cantidad Registros: 3
[INFO] Registro insertado en sector: Disco\Plato_3\Superficie_0\Pista_0\Sector_0.txt
Tama├▒o de registro: 206
Usando nuevo bloque con espacio: 1532
[INFO] Registro insertado en sector: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
Carga finalizada. Registros insertados: 3 / 3. Bloques utilizados: 1
Capacidad total en bytes: 65536
Capacidad Disponible: 64659
Capacidad Ocupada: 877
```

```
Cantidad Registros: 7
[INFO] Registro insertado en sector: Disco\Plato_3\Superficie_0\Pista_0\Sector_0.txt
Tama├▒o de registro: 206
Usando nuevo bloque con espacio: 1532
[INFO] Registro insertado en sector: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_0\Pista_0\Sector_1.txt
Nuevo bloque asignado para la relaci├▒n
Carga finalizada. Registros insertados: 7 / 7. Bloques utilizados: 2
Capacidad total en bytes: 65536
Capacidad Disponible: 63535
Capacidad Ocupada: 2001
```

```
_________914
________-1
_________1#_________0#_________3#Braund, Mr. Owen Harris_____________________________
_________2#_________1#_________1#Cumings, Mrs. John Bradley (Florence Briggs Thayer)_____
```

Disco > Plato_0 > Superficie_1 > Pista_0 > Sector_1.txt

```
1  __________0
2  ________-1
3  _________1#_________0#_________3#Braund, Mr. Owen Harris_____________________________
4  _________2#_________1#_________1#Cumings, Mrs. John Bradley (Florence Briggs Thayer)_____
5
```



Espacio Disponible	1532
Posicion Eliminado	-1

longitud fija

```
bool insertarRegistro(Bloque &bloque, const string &registro, int tam, int& tam_sector) {
```

```
string linea = bloque.get_cabecera();
Cabecera_fija c(linea);
```

```
if (c.inicio_lista_eliminada != -1) {
    // Hay un espacio libre en la lista de eliminados
    int tam_disponible = bloque.get_espacioDisponible();
    int pos = c.inicio_lista_eliminada;
    string contenido = bloque.getContenido_en_posicion(pos, registro.size());

    int siguiente;
    try {
        string campo = contenido.substr(1, 10); // Extraer 10 caracteres
        string limpio;
        for (char c : campo) {
            if (isdigit(c)) limpio += c;
            else if (c == '-') limpio += c;
        }
        siguiente = limpio.empty() ? -1 : stoi(limpio);
    } catch (...) {
        siguiente = -1;
    }

    stringstream ss;
    ss << setw(10) << setfill('_') << siguiente;
    int basura;
    bloque.set_espacioDisponible(tam_disponible - tam);
    bloque.setContenido_en_posicion(pos, registro, registro.size(), basura);
    bloque.set_cabecera(ss.str());
    cout << "[INFO] Registro insertado en posición reutilizada: " << pos << endl;
} else {
```

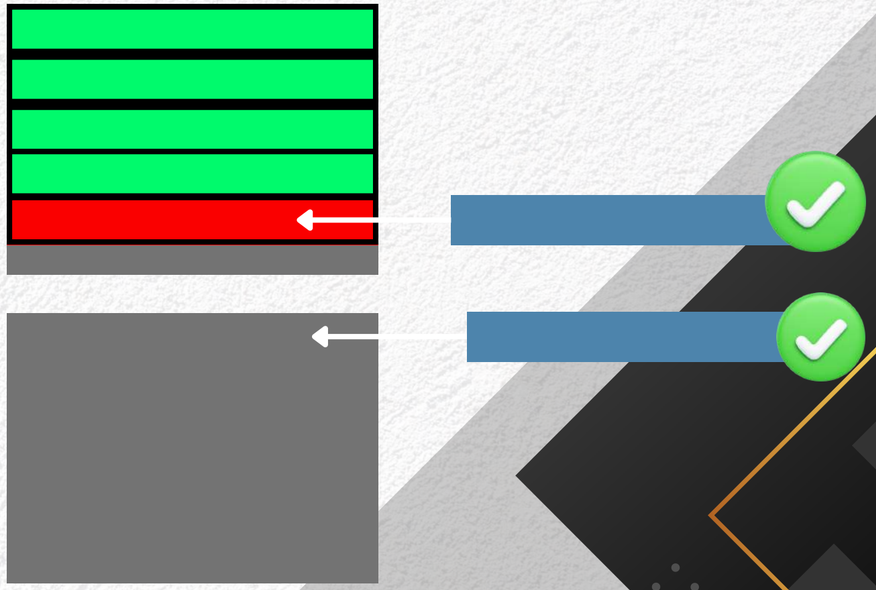
```
bool insertarRegistro(Bloque &bloque, const string &registro, int tam, int& tam_sector) {
    } else {
        if (tam <= 0 || tam > tam_disponible) {

        int cantidad_sectores = bloque.getCantidad_Sectores();
        int registros_por_sector = tam_sector / tam;
        if (registros_por_sector == 0) {
            bloque.set_espacioDisponible(0);
            return false;
        }

        int registros_totales = cantidad_sectores * registros_por_sector;
        int registros_ocupados = (cantidad_sectores * tam_sector - tam_disponible) / tam;
        if (registros_ocupados >= registros_totales) {
            bloque.set_espacioDisponible(0);
            return false;
        }

        int sector_objetivo = registros_ocupados / registros_por_sector;
        if (sector_objetivo >= cantidad_sectores) {
            bloque.set_espacioDisponible(0);
            return false;
        }

        bloque.addContenido(sector_objetivo, registro);
        bloque.set_espacioDisponible(tam_disponible - tam);
        cout << "[INFO] Registro insertado en sector: " << bloque.getSector(sector_objetivo) << endl;
    }
```

Espacio Disponible	1532
Posicion Eliminado	-1

```
void contenido_parcial(fstream& file, string cabecera, DiscoManager &manager,
                       int sectoresPorBloque, int& tam_sector, string nombre, int N) {
```

```
bool insertado = false;
for (auto& bloque : bloques_relacion) {
    if (insertarRegistro(bloque, linea_formateada, tam_registro, tam_sector)) {
        insertado = true;
        bloque_actual = &bloque;
        registros_insertados++;
        break;
    }
}
```

```
if (!insertado && registros_insertados < N) {
    Bloque nuevo_bloque = manager.Ubicar_BloqueVacio(nombre,"fija");
    if (nuevo_bloque.get_espacioDisponible() >= tam_registro) {
        bloques_relacion.push_back(nuevo_bloque);
        bloque_actual = &bloques_relacion.back();
        insertarRegistro(*bloque_actual, linea_formateada, tam_registro, tam_sector);
        registros_insertados++;
        cout << "Nuevo bloque asignado para la relación" << endl;
    } else {
        cerr << "Error: No hay espacio suficiente en disco" << endl;
        break;
    }
}
```



longitud variable

```
Cantidad de Archivos: 20
[INFO] Registro insertado en sector: Disco\Plato_3\Superficie_0\Pista_0\Sector_0.txt
Usando nuevo bloque con espacio: 1532
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_2\Superficie_0\Pista_0\Sector_1.txt
Nuevo bloque asignado para la relacion
[INFO] Registro insertado nuevo en: Disco\Plato_2\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_2\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_2\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_2\Superficie_0\Pista_0\Sector_1.txt
Carga parcial finalizada. Registros procesados: 20, Bloques utilizados: 2
Capacidad total en bytes: 65536
Capacidad Disponible: 63435
```

Disco > Plato_0 > Superficie_1 > Pista_0 > Sector_1.txt

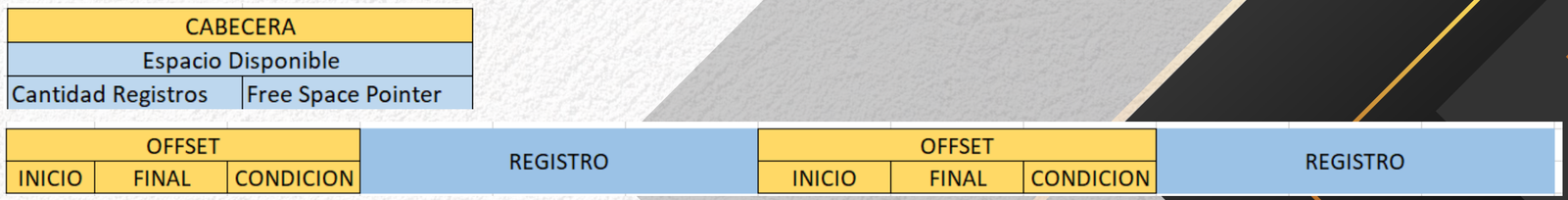
```
1   _________138
2   _________15_______458
3   _________0|________59|01#0#3#Braund, Mr. Owen Harris#male#22#1#0#A/5 21171#7.25##S
4   _________83|_______177|02#1#1#Cumings, Mrs. John Bradley (Florence Briggs Thayer)#female#38#1#0#PC 175
5   ________201|_______269|03#1#3#Heikkinen, Miss. Laina#female#26#0#0#STON/O2. 3101282#7.925##S
6   ________293|_______376|04#1#1#Futrelle, Mrs. Jacques Heath (Lily May Peel)#female#35#1#0#113803#53.1#C
7   ________400|_______457|05#0#3#Allen, Mr. William Henry#male#35#0#0#373450#8.05##S
```

Disco > Plato_1 > Superficie_0 > Pista_0 > Sector_1.txt

```
1   ________481|_______530|06#0#3#Moran, Mr. James#male##0#0#330877#8.4583##Q
2   _________73|_______134|07#0#1#McCarthy, Mr. Timothy J#male#54#0#0#17463#51.8625#E46#S
3   ________158|_______222|08#0#3#Palsson, Master. Gosta Leonard#male#2#3#1#349909#21.075##S
4   ________246|_______333|09#1#3#Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)#female#27#0#2#347742#1
5   ________357|_______431|010#1#2#Nasser, Mrs. Nicholas (Adele Achem)#female#14#1#0#237736#30.0708##C
```

Disco > Plato_1 > Superficie_1 > Pista_0 > Sector_1.txt

```
1   ________455|_______524|011#1#3#Sandstrom, Miss. Marguerite Rut#female#4#1#1#PP 9549#16.7#G6#S
2   _________93|_______158|012#1#1#Bonnell, Miss. Elizabeth#female#58#0#0#113783#26.55#C103#S
3   ________182|_______249|013#0#3#Saundercock, Mr. William Henry#male#20#0#0#A/5. 2151#8.05##S
4   ________273|_______336|014#0#3#Andersson, Mr. Anders Johan#male#39#1#5#347082#31.275##S
5   ________360|_______434|015#0#3#Vestrom, Miss. Hulda Amanda Adolfina#female#14#0#0#350406#7.8542##S
6
```



longitud variable

```
bool insertarRegistro_Var(Bloque &bloque, const string &registro, int tam_registro, int& tam_sector)
```

```
for (int i = 0; i < bloque.getCantidad_Sectores(); ++i) {
    string &contenido = bloque.getContenidoRef(i);
    size_t pos = 0;
    while (pos + metadata_size <= contenido.size()) {
        string metadato = contenido.substr(pos, metadata_size);
        if (metadato[22] == '1') { // Registro eliminado

            // Obtener inicio y fin del registro eliminado
            string ini_str = metadato.substr(0, 10);
            string fin_str = metadato.substr(11, 10);
            int ini = limpiar_numero(ini_str);
            int fin = limpiar_numero(fin_str);
            int espacio_registro = (fin - ini + 1) + metadata_size;

            if (espacio_registro >= tam_total) {
                // Sobrescribir el contenido eliminado
                stringstream ss;
                ss << setw(10) << setfill('_') << ini << "|"
                   << setw(10) << setfill('_') << (ini + tam_registro - 1) << "|"
                   << "0" << registro;

                contenido.replace(pos, ss.str().length(), ss.str());

                cabecera.cantidad++;
                cabecera.siguiente_libre=contenido.size();  // Actualiza con el final actual
                bloque.set_cabecera(cabecera.to_string());
                bloque.set_espacioDisponible(bloque.get_espacioDisponible() - tam_total);
```

```
for (int i = 0; i < bloque.getCantidad_Sectores(); ++i) {
    string &contenido = bloque.getContenidoRef(i);
    int disponible = tam_sector - contenido.size();
    if (disponible >= tam_total) {
        int inicio = cabecera.siguiente_libre;
        int fin = inicio + tam_registro - 1;

        stringstream ss;
        ss << setw(10) << setfill('_') << inicio << "|"
           << setw(10) << setfill('_') << fin << "|"
           << "0" << registro;

        contenido += ss.str();

        cabecera.cantidad++;
        cabecera.siguiente_libre=contenido.size();  // Posición siguiente libre actualizada
        bloque.set_cabecera(cabecera.to_string());
        bloque.set_espacioDisponible(bloque.get_espacioDisponible() - tam_total);

        cout << "[INFO] Registro insertado nuevo en: " << bloque.getSector(i) << endl;
        return true;
    }
}
```

204								
3	455							
0	59	0	Registro			83	177	1
Registro			201	269	0	Registro		
293	376	1	Registro					

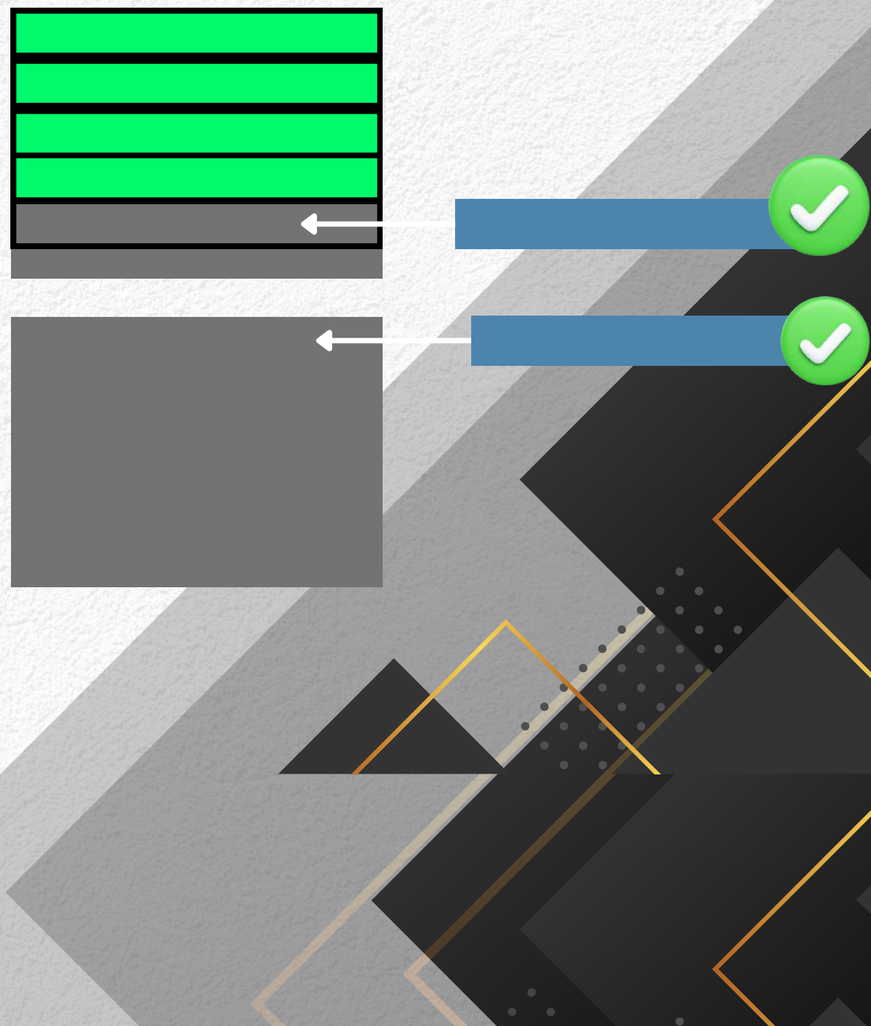
204								
10	455							
0	59	0	Registro			83	177	0
Registro			201	269	0	Registro		
			Registro					

longitud variable

```
void contenido_parcial_var(fstream& file, string cabecera, DiscoManager &manager,
                           int sectoresPorBloque, int& tam_sector, string nombre, int N) {
```

```
// Intentar insertar en bloques existentes primero
bool insertado = false;
for (auto& bloque : bloques_relacion) {
    if (insertarRegistro_Var(bloque, linea_formateada, tam_registro, tam_sector)) {
        insertado = true;
        bloque_actual = &bloque;
        break;
    }
}
```

```
if (!insertado) {
    Bloque nuevo_bloque = manager.Ubicar_BloqueVacio(nombre,"variable");
    if (nuevo_bloque.get_espacioDisponible() >= tam_registro) {
        bloques_relacion.push_back(nuevo_bloque);
        bloque_actual = &bloques_relacion.back();
        insertarRegistro_Var(*bloque_actual, linea_formateada, tam_registro,tam_sector);
        cout << "Nuevo bloque asignado para la relacion" << endl;
    } else {
        cerr << "Error: No hay espacio suficiente en disco" << endl;
        break;
    }
```



Sectores Ocupados

```
14. Mostrar Sectores Ocupados
15. Ingresar 1 Registro L.Variable
16. Ingresar 1 Registro L.Fija
Seleccione una opcion: 14
Sectores Ocupados:
(0,0,0,0)
(0,1,0,0)
(1,0,0,0)

(1,1,0,0)
(2,0,0,0)
(2,1,0,0)

(3,0,0,0)
(3,1,0,0)
(0,0,0,1)

(0,1,0,1)
(1,0,0,1)
(1,1,0,1)

(2,0,0,1)
(2,1,0,1)
(3,0,0,1)
```

```
Seleccione una opcion: 4
Nombre Archivo: Titanic
Cantidad Registros: 10
[INFO] Registro insertado en sector: Disco\Plato_3\Superficie_0\Pista_0\Sector_0.txt
Tama├o de registro: 206
Usando nuevo bloque con espacio: 1532
[INFO] Registro insertado en sector: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_1\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_0\Pista_0\Sector_1.txt
Nuevo bloque asignado para la relaci├n
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado en sector: Disco\Plato_2\Superficie_1\Pista_0\Sector_1.txt
Carga finalizada. Registros insertados: 10 / 10. Bloques utilizados: 2
Capacidad total en bytes: 65536
Capacidad Disponible: 62917
Capacidad Ocupada: 2619
```

```
class DiscoManager {
void mostrar_BloqueVacio();
    void mostrar_SectoresOcupados() {
        cout << "Sectores Ocupados: " << endl;
        for (int i = 0; i < Diccionario.size(); i++) {
            int index = 0;
            for (int j = 0; j < Diccionario[i].cantidad_sectores(); j++) {
                int cantidadReg = Diccionario[i].get_cant_reg_Sector(j);
                for (int k = 0; k < cantidadReg; k++, index++) {
                    if (Diccionario[i].getNombre(index) != "NULL") {
                        Coord cIni = Diccionario[i].getCordsIni(index);
                        Coord cFin = Diccionario[i].getCordsFin(index);
                        int plato = cIni.plato;
                        int superficie = cIni.superficie;
                        int sector = cIni.sector;
                        int pista = cIni.pista;

                        while (true) {
                            cout << "(" << plato << "," << superficie << "," << pista << "," << sectc
                            //verificar que ya se llego al final
                            if (plato == cFin.plato &&
                                superficie == cFin.superficie &&
                                sector == cFin.sector &&
                                pista == cFin.pista) {
                                break;
                            }

                            avanzarPosicion(plato, superficie, sector, pista, disco);
```

Eliminar Fijo

```
__________0
________-1
________1#________0#________3#Braund, Mr. Owen Harris_____________________________________
_________2#________1#________1#Cumings, Mrs. John Bradley (Florence Briggs Thayer)_________
```

```
===== Menu Principal =====
1. Crear disco
2. Select From * L.Fija
3. Cargar todo CSV L.fija
4. Agregar N registros L.fija
5. Salir
6. Select From Where
7. Eliminar un registro
8. Mostrar Bloques Libres
9. Mostrar Bloques Ocupados
10. Cargar todo CSV L.Variable
11. Agregar N registros L.Variable
12. SELECT FROM* L.Variable
13. Eliminar un registo L.Variable
14. Mostrar Sectores Ocupados
15. Ingresar 1 Registro L.Variable
16. Ingresar 1 Registro L.Fija
Seleccione una opcion: 7
Nombre Archivo: Titanic
Comparador: =
Atributo: PassengerId
Valor: 2
[INFO] Registro eliminado en : Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
```

0
-1
Registro
Registro

206
1
Registro
*____-1Registro

Disco > Plato_0 > Superficie_1 > Pista_0 > Sector_1.txt

```
1   _________206
2   _________1
3   _________1#________0#________3#Braund, Mr. Owen Harris_____________________________________
4   *________-1_____________________________________________________________________________
5
```

Eliminar Variable

Disco > Plato_0 > Superficie_1 > Pista_0 > Sector_1.txt

```
1   _________596
2   ________10_______455
3   _________0|________59|01#0#3#Braund, Mr. Owen Harris#male#22#1#0#A/5 21171#7.25##S
4   ________83|_______177|02#1#1#Cumings, Mrs. John Bradley (Florence Briggs Thayer)#female#38#1#0#PC 175
5   _______201|_______269|03#1#3#Heikkinen, Miss. Laina#female#26#0#0#STON/O2. 3101282#7.925##S
6   _______293|_______376|04#1#1#Futrelle, Mrs. Jacques Heath (Lily May Peel)#female#35#1#0#113803#53.1#C
7   _______400|_______457|05#0#3#Allen, Mr. William Henry#male#35#0#0#373450#8.05##S
8
```

```
[INFO] Registro insertado en sector: Disco\Plato_3\Superficie_0\Pista_0\Sector_0.txt
Usando nuevo bloque con espacio: 1532
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
[INFO] Registro insertado nuevo en: Disco\Plato_1\Superficie_0\Pista_0\Sector_1.txt
Carga parcial finalizada. Registros procesados: 10, Bloques utilizados: 1
Capacidad total en bytes: 65536
```

```
10. Cargar todo CSV L.Variable
11. Agregar N registros L.Variable
12. SELECT FROM* L.Variable
13. Eliminar un registo L.Variable
14. Mostrar Sectores Ocupados
15. Ingresar 1 Registro L.Variable
16. Ingresar 1 Registro L.Fija
Seleccione una opcion: 13
Nombre Archivo: Titanic
Comparador: =
Atributo: PassengerId
Valor: 3
[INFO] Registro eliminado en: Disco\Plato_0\Superficie_1\Pista_0\Sector_1.txt
```

204							
3	455						
0	59	0	Registro			83	177	1
Registro			201	269	0	Registro		

309							
2	455						
0	59	0	Registro			83	177	1
Registro			201	269	0	Registro		

Disco > Plato_0 > Superficie_1 > Pista_0 > Sector_1.txt

```
1   _________688
2   _________9_______455
3   _________0|________59|01#0#3#Braund, Mr. Owen Harris#male#22#1#0#A/5 21171#7.25##S
4   ________83|_______177|02#1#1#Cumings, Mrs. John Bradley (Florence Briggs Thayer)#female#38#1#0#PC 175
5   _______201|_______269|13#1#3#Heikkinen, Miss. Laina#female#26#0#0#STON/O2. 3101282#7.925##S
6   _______293|_______376|04#1#1#Futrelle, Mrs. Jacques Heath (Lily May Peel)#female#35#1#0#113803#53.1#C
7   _______400|_______457|05#0#3#Allen, Mr. William Henry#male#35#0#0#373450#8.05##S
8
```

Diccionario

```
class DiscoManager {
vector<Metadata> Diccionario;
```

Metadata	SCHEMAS	Titanic	NULL	NULL	NULL	NULL	NULL
Sector Inicial 0,0,0,0	1,1,0,0	3,0,0,0	0,1,0,1	...	...	...	
Sector Final 1,0,0,0	2,1,0,0	0,0,0,1	1,1,0,1	...	...	...	

```
void cargarDiccionario() {
```

```
for (size_t i = 0; i < lineas.size(); ++i) {
    if (lineas[i].substr(0, 8) == "Metadata") {
        if (i + 2 >= lineas.size()) {
            cerr << "Coordenadas incompletas para Metadata en línea " << i << endl;
            continue;
        }

        // --- Coordenadas de inicio ---
        string inicio = lineas[i + 1];
        int pl_ini = stoi(inicio.substr(0, 10));
        int sup_ini = stoi(inicio.substr(10, 10));
        int pis_ini = stoi(inicio.substr(20, 10));
        int sec_ini = stoi(inicio.substr(30, 10));

        // --- Coordenadas de fin ---
        string fin = lineas[i + 2];
        int pl_fin = stoi(fin.substr(0, 10));
        int sup_fin = stoi(fin.substr(10, 10));
        int pis_fin = stoi(fin.substr(20, 10));
        int sec_fin = stoi(fin.substr(30, 10));
```

```
Bloque Ubicar_BloqueVacio(const string& nombreRelacion, string opcion) {
    Bloque bloque;
    int nuevaInicio[4] = {-1, -1, -1, -1};
    int nuevaFin[4] = {-1, -1, -1, -1};
    bool encontrado = false;

    for (int i = 0; i < Diccionario.size() && !encontrado; i++) {
        int index = 0;
        for (int j = 0; j < Diccionario[i].cantidad_sectores() && !encontrado; j++) {
            int cantidadReg = Diccionario[i].get_cant_reg_Sector(j);
            for (int k = 0; k < cantidadReg && !encontrado; k++) {
                if (Diccionario[i].getNombre(index) == "NULL") {
                    Coord cIni = Diccionario[i].getCordsIni(index);
                    Coord cFin = Diccionario[i].getCordsFin(index);

                    nuevaInicio[0] = cIni.plato;
                    nuevaInicio[1] = cIni.superficie;
                    nuevaInicio[2] = cIni.pista;
                    nuevaInicio[3] = cIni.sector;

                    nuevaFin[0] = cFin.plato;
                    nuevaFin[1] = cFin.superficie;
                    nuevaFin[2] = cFin.pista;
                    nuevaFin[3] = cFin.sector;
```

Diccionario

Disco > Plato_0 > Superficie_0 > Pista_0 > Sector_0.txt

```
49__________
Metadata________
0_________0_________0_________0_________
1_________0_________0_________0_________
Metadata________
1_________1_________0_________0_________
2_________1_________0_________0_________
Schemas_________
3_________0_________0_________0_________
0_________0_________0_________1_________
Housing_________
0_________1_________0_________1_________
1_________1_________0_________1_________
NULL____________
2_________0_________0_________1_________
3_________0_________0_________1_________
NULL____________
3_________1_________0_________1_________
0_________1_________0_________2_________
NULL____________
1_________0_________0_________2_________
2_________0_________0_________2_________
NULL____________
2_________1_________0_________2_________
3_________1_________0_________2_________
NULL____________
0_________0_________0_________3_________
1_________0_________0_________3_________

```

Bloque

```cpp
class Bloque {
private:
    vector<string> contenido;
    vector<string> referencias;
    size_t espacio_disponible = 0;
    string cabecera;
public:
    Bloque() = default;

    void agregarSector(const string& s) {…

    string getContenido(int i) const {…

    string getContenidoTotal() {…

    void limpiar() {…

    string getSector(int i) {…

    int get_espacioDisponible() {…

    void set_espacioDisponible(int nuevo) {…

    void set_contenido(string cont, int i) {…

    int getCantidad_Sectores() {…

    void addContenido(int i, const string& s) {…
```

Bloque > Housing0.txt

```
1   ________392
2   ________-1
3   __13300000#_____7420#________4#________2#________3#yes#no_#no_#no_#yes#________2#yes#furnished__
4   __12250000#_____8960#________4#________4#________4#yes#no_#no_#no_#yes#________3#no_#furnished__
5   __12250000#_____9960#________3#________2#________2#yes#no_#yes#no_#no_#________2#yes#semi-furnis
6   __12215000#_____7500#________4#________2#________2#yes#no_#yes#no_#yes#________3#yes#furnished__
7   __11410000#_____7420#________4#________1#________2#yes#yes#yes#no_#yes#________2#no_#furnished__
8   __10850000#_____7500#________3#________3#________1#yes#no_#yes#no_#yes#________2#yes#semi-furnis
9   __10150000#_____8580#________4#________3#________4#yes#no_#no_#no_#yes#________2#yes#semi-furnis
10  __10150000#____16200#________5#________3#________2#yes#no_#no_#no_#no_#________0#no_#unfurnished
11  ___9870000#_____8100#________4#________1#________2#yes#yes#yes#no_#yes#________2#yes#furnished__
12  ___9800000#_____5750#________3#________2#________4#yes#yes#no_#no_#yes#________1#yes#unfurnished
13
```

```
_________596
________10_______455
_________0|________59|01#0#3#Braund, Mr. Owen Harris#male#22#1#0#A/5 21171#7.25##S
________83|_______177|02#1#1#Cumings, Mrs. John Bradley (Florence Briggs Thayer)#female#38#1#0#PC 175
_______201|_______269|03#1#3#Heikkinen, Miss. Laina#female#26#0#0#STON/O2. 3101282#7.925##S
_______293|_______376|04#1#1#Futrelle, Mrs. Jacques Heath (Lily May Peel)#female#35#1#0#113803#53.1#C
_______400|_______457|05#0#3#Allen, Mr. William Henry#male#35#0#0#373450#8.05##S
_______481|_______530|06#0#3#Moran, Mr. James#male##0#0#330877#8.4583##Q
________73|_______134|07#0#1#McCarthy, Mr. Timothy J#male#54#0#0#17463#51.8625#E46#S
_______158|_______222|08#0#3#Palsson, Master. Gosta Leonard#male#2#3#1#349909#21.075##S
_______246|_______333|09#1#3#Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)#female#27#0#2#347742#1
_______357|_______431|010#1#2#Nasser, Mrs. Nicholas (Adele Achem)#female#14#1#0#237736#30.0708##C
```

Bloque

```cpp
struct Cabecera_Var {
    int cantidad;           // número de registros
    int siguiente_libre;    // posición para insertar el próximo registro

    Cabecera_Var() : cantidad(0), siguiente_libre(0) {}

    Cabecera_Var(const string& entrada) {
        if (entrada.size() >= 20) {
            string parte1 = entrada.substr(0, 10);
            string parte2 = entrada.substr(10, 10);

            parte1.erase(remove(parte1.begin(), parte1.end(), '_'), parte1.end());
            parte2.erase(remove(parte2.begin(), parte2.end(), '_'), parte2.end());
```

```cpp
struct Cabecera_fija {
    int inicio_lista_eliminada;

    Cabecera_fija() : inicio_lista_eliminada(-1) {}

    Cabecera_fija(const string& entrada) {
        string limpio;
        for (char c : entrada) {
            if (isdigit(c) || c == '-') {
                limpio += c;
            }
        }
        try {
            inicio_lista_eliminada = stoi(limpio);
        } catch (...) {
            inicio_lista_eliminada = -1;
        }
    }
};
```

Bloque Libre y ocupado

```
5. Salir
6. Select From Where
7. Eliminar un registro
8. Mostrar Bloques Libres
9. Mostrar Bloques Ocupados
10. Cargar todo CSV L.Variable
11. Agregar N registros L.Variable
12. SELECT FROM* L.Variable
13. Eliminar un registo L.Variable
14. Mostrar Sectores Ocupados
15. Ingresar 1 Registro L.Variable
16. Ingresar 1 Registro L.Fija
Seleccione una opcion: 8
Bloque Libre
NULL
Cordenas Sectores:
3,1,0,1
0,1,0,2
Capacidad disponible:
1536

Bloque Libre
NULL
Cordenas Sectores:
1,0,0,2
2,0,0,2
Capacidad disponible:
1536
```

```
16. Ingresar 1 Registro L.Fija
Seleccione una opcion: 9
Bloque Ocupado
Metadata
Cordenas Sectores:
0,0,0,0
1,0,0,0
Capacidad disponible:
49_________

Bloque Ocupado
Metadata
Cordenas Sectores:
1,1,0,0
2,1,0,0
Capacidad disponible:
49_________

Bloque Ocupado
Schemas
Cordenas Sectores:
3,0,0,0
0,0,0,1
Capacidad disponible:
________1030

Bloque Ocupado
Housing
Cordenas Sectores:
0,1,0,1
1,1,0,1
Capacidad disponible:
_________392
```

Bloque Libre y ocupado(código)

```
void mostrar_BloqueVacio(){
    for (int i = 0; i < Diccionario.size(); i++) {
        int index = 0;
        for (int j = 0; j < Diccionario[i].cantidad_sectores(); j++) {
            int cantidadReg = Diccionario[i].get_cant_reg_Sector(j);
            for (int k = 0; k < cantidadReg; k++, index++) {
                if (Diccionario[i].getNombre(index) == "NULL") {
                    cout<<"Bloque Libre"<<endl;
                    cout<<Diccionario[i].getNombre(index)<<endl;
                    cout<<"Cordenas Sectores:"<<endl;
                    Coord cIni = Diccionario[i].getCordsIni(index);
                    Coord cFin = Diccionario[i].getCordsFin(index);
                    cout<<cIni.plato<<","<<cIni.superficie<<","<<cIni.pista<<","<<cIni.sector<<endl;
                    cout<<cFin.plato<<","<<cFin.superficie<<","<<cFin.pista<<","<<cFin.sector<<endl;
                    cout<<"Capacidad disponible: "<<endl;
                    cout<<sectoresPorBloque*tam_sector<<endl;
                    cout<<endl;
                }
            }
        }
    }
}
```

SELECT FROM*

```
5. Salir
6. Select From Where
7. Eliminar un registro
8. Mostrar Bloques Libres
9. Mostrar Bloques Ocupados
10. Cargar todo CSV L.Variable
11. Agregar N registros L.Variable
12. SELECT FROM* L.Variable
13. Eliminar un registo L.Variable
14. Mostrar Sectores Ocupados
15. Ingresar 1 Registro L.Variable
16. Ingresar 1 Registro L.Fija
Seleccione una opcion: 2
Nombre Archivo: Housing
price area bedrooms bathrooms stories mainroad guestroom basement hotwaterhea
ingstatus
13300000 7420 4 2 3 yes no no no yes 2 yes furnished
 12250000 8960 4 4 4 yes no no no yes 3 no furnished
 12250000 9960 3 2 2 yes no yes no no 2 yes semi-furnished
 12215000 7500 4 2 2 yes no yes no yes 3 yes furnished
 11410000 7420 4 1 2 yes yes yes no yes 2 no furnished
 10850000 7500 3 3 1 yes no yes no yes 2 yes semi-furnished
 10150000 8580 4 3 4 yes no no no yes 2 yes semi-furnished
 10150000 16200 5 3 2 yes no no no no 0 no unfurnished
 9870000 8100 4 1 2 yes yes yes no yes 2 yes furnished
 9800000 5750 3 2 4 yes yes no no yes 1 yes unfurnished
```

```
void Select_Todo(string cabecera, vector<Bloque>& Relacion) {
    imprimir_cabecera(cabecera);

    for (int i = 0; i < Relacion.size(); ++i) {
        for (int j = 0; j < Relacion[i].getCantidad_Sectores(); ++j) {
            string contenido = Relacion[i].getContenido(j);
```

```
if (registro.empty() || registro[0] == '*') continue;

string limpio;
for (char c : registro) {
    if (c != '_') limpio += c;
}

vector<string> campos;
string campo_actual;
for (char c : limpio) {
    if (c == '#') {
        campos.push_back(campo_actual);
        campo_actual.clear();
    } else {
        campo_actual += c;
    }
}
if (!campo_actual.empty()) campos.push_back(campo_actual);

for (const string& campo : campos) {
    cout << campo << " ";
}
```

REFERENCIAS

- Silberschatz, A., Korth, H. F., & Sudarshan, S. (2011). Database system concepts (6th ed.). McGraw-Hill.

- Garcia-Molina, H., Ullman, J. D., & Widom, J. (2008). Database systems: The complete book (2nd ed.). Pearson Education.