MANUAL TECNICO

Base de datos

BASE DE DATOS

ESTRUCTURAS UTILIZADAS PARA BASE DE DATOS

```
struct informacionDato {
    char tipoChar[sizeChar];
    string tipoString;
    int tipoEntero;
    double tipoDouble;
    int esAlta;
};
typedef struct informacionDato datosColumnasTabla;

struct tipotablaH {
    informacionDato *tabla[sizeTablaHash];
    int elementos;
    double factorcarga;
};
typedef struct tipotablaH tablaHash;

struct nombreColumnas {
    string nombreC;
    string tipoDatoC;
    informacionDato *valorFila[sizeDatos];
};
```

```
typedef struct nombreColumnas Columnas;

struct nombreTablas {
    Columnas *nombreColumna[sizeColumnas];
    string nombreTabla;
};
typedef struct nombreTablas Tablas;


MODELO TABLA HASH
void CrearTabla(tablaHash *t) {
    int j;
    for (j = 0; j < sizeTablaHash; j++) {
        t->tabla[j] == NULL;
        cout << "\n POsicion de tabla " << j;
    }
    t->elementos = 0;
    t->factorcarga = 0.0;
}

/* transforma los caracteres de la clave en valores enteros*/
long transforma(char *clave) {
    int j;
    long d = 0;
    for (j = 0; j < strlen(clave); j++) {
        d = d + clave[j];
    }
    cout << "\n Clave " << d;
    return ((d >= 0) ? d : -d);
```

```
    /* dirección recibe la tabladispersa y la clave para colocar
esta ultima en la tabla*/
}

int direccion(tablaHash *t, char *clave) {
    int i = 0;
    long p = 0, d = 0;
    for (int j = 1; j < strlen(clave); j++) {
        d += (int) clave[j];
    }
    cout << "\nClave " << d;
    p = d % sizeTablaHash;
    cout << "\n Direccion hash " << p;
    return (int) p;
}

/* Inserta los datos que representa la clave en la tabla hash*/
void insertar(tablaHash *t, datosColumnasTabla r, int p) {
    datosColumnasTabla *pr;
    int posicion;
    pr = new (struct informacionDato);
    strcpy(pr->tipoChar, r.tipoChar);
    pr->esAlta = 1;
    posicion = p;
    t->tabla[posicion] = pr;
    t->elementos++;
    t->factorcarga = (t->elementos) / sizeTablaHash;
    if (t->factorcarga > 0.7) {
```

```
        puts("\nFactor de Carga supera el 70% de la tabla");
    }
}

/* Busca el elemento en la tabla e imprime si lo encuentra o
no ...
 ****** como la clave es igual a otra entonces buscar que
este en ese
 listado con la misma cadena */
datosColumnasTabla *buscar(tablaHash *t, char *clave) {
    datosColumnasTabla *pr;
    int posicion;
    posicion = direccion(t, clave);
    pr = t->tabla[posicion];
    if (pr != NULL) {
        if (!(pr->esAlta)) {
            pr = NULL;
        }
    }
    return pr;
}

/* Elimina el elemento de la tabla hash */
int eliminar(tablaHash *t, char * clave) {
    int posicion;
    posicion = direccion(t, clave);
    if (t->tabla[posicion] != NULL) {
        t->tabla[posicion] -> esAlta = 0;
    } else
```

```
        return 1;
}
```

URL de git:
/home/esmeralda/Documentos/ED/.git