

# Proyecto Prog III

Estadisticas Futbol

---

**Di Tomaso, Pappalardo,  
Buhler**

**Agustin, Mateo, Augusto**

**Prof. Eliezer Salcedo**

# Mejorar rendimiento sistema estadisticas

## Objetivos

- Reducir los tiempos de ejecucion del sistema.
- Facilidad de uso para el cliente.
- Uso apropiado de las estuctruas y algortimos de datos

# Analisis del código

## ¿Como logramos mejorar el sistema de estadisticas de futbol?

- Utilizacion de Hashmaps:
  - Velocidad de acceso promedio constante.
  - Flexibilidad en el uso de claves.
- Utilizacion de Arboles:
  - Ordenamiento automático.
  - Rendimiento consistente en el peor de los casos.
- Validaciones Estructuradas.
  - Claridad y Organizacion del codigo.
  - Facil mantenimiento y escalabilidad
  - Reutilizacion de codigo
  - Manejo de errores y flujo de control
- Utilizacion de librerias estandares optimizadas.
  - regex
  - algorithm
  - limits
  - map

```
std::unordered_map<std::string, std::set<Partido> > partidosPorCompeticion;
std::unordered_map<std::string, std::unordered_map<std::string, int > > golesAFavor; // equipo -> competicion
std::unordered_map<std::string, std::unordered_map<std::string, int > > golesEnContra; // equipo -> competicion

std::unordered_map<std::string, std::unordered_map<std::string, double > > promedioGolesAFavor; // competicion
std::unordered_map<std::string, std::unordered_map<std::string, double > > promedioGolesEnContra; // competicion
std::unordered_map<std::string, std::unordered_map<std::string, int > > partidosJugados; // equipo -> competicion
std::unordered_map<std::string, int > golesPorCompeticion; // competicion -> total de goles
std::unordered_map<std::string, std::unordered_map<std::string, int > > triunfos; // equipo -> competicion
std::unordered_map<std::string, std::unordered_map<std::string, int > > derrotas; // equipo -> competicion
std::unordered_map<std::string, std::unordered_map<std::string, std::unordered_map<std::string, int > > > golesPorEquipoPorCompeticion;
std::unordered_map<std::string, std::unordered_map<std::string, int > > golesPorEquipoPorCompeticion; // competicion -> golesPorEquipoPorCompeticion
```

```
switch (opcion) {
    case 1: {
        std::string equipo, competicion;

        std::cout << "Ingrese el nombre del equipo: ";
        std::cin.ignore(); // Limpiar cualquier salto de linea pendiente en el buffer
        std::getline( & std::cin, & equipo);
        while (equipo.empty()) { // Validación para que no esté vacío
            std::cout << "Entrada inválida. Intente nuevamente.\nIngrese el nombre del equipo: ";
            std::getline( & std::cin, & equipo);
        }

        std::cout << "Ingrese el nombre de la competición: ";
        std::getline( & std::cin, & competicion);
        while (competicion.empty()) { // Validación para que no esté vacío
            std::cout << "Entrada inválida. Intente nuevamente.\nIngrese el nombre de la competición: ";
            std::getline( & std::cin, & competicion);
        }

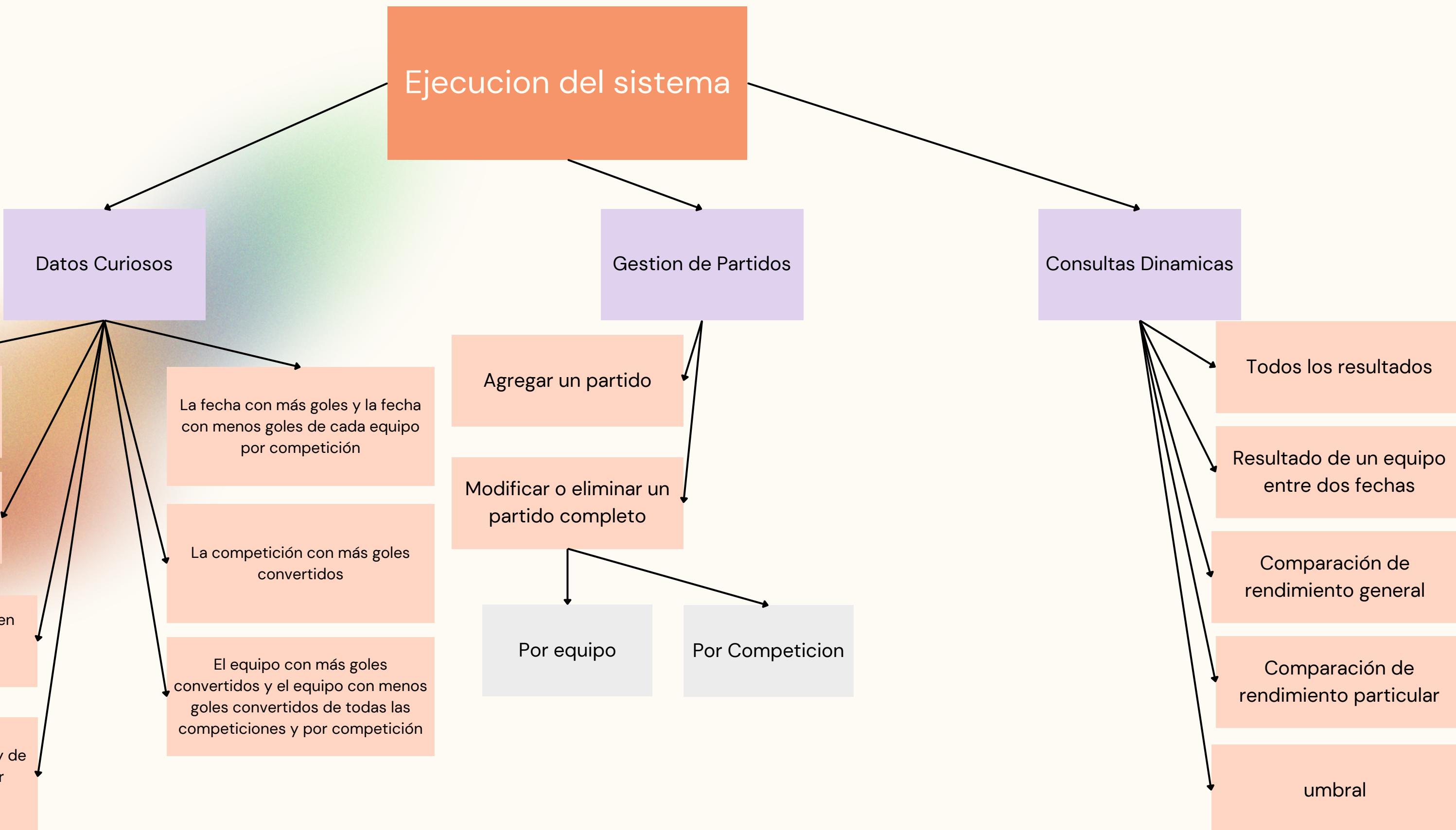
        consultaResultadosEquipoCompeticion(equipo, competicion);
        break;
    }
}
```

# Analisis Estructural

## Explicacion del uso de flujo del sistema

- Es un sistema robusto y facil de utilizar.
- Posee 3 instancias principales y una opcion de salida.
  - Estadisticas Curiosas
  - Gestion de Partidos
  - Consultas Dinamicas
  - Salir
- Estadisticas Curiosas
  - Posee 7 opciones interactivas y una vuelta al menu principal
  - Filtra determinadas estadisticas para el usuario a traves de diferentes funciones
- Gestion de Partidos
  - Posee 2 opciones y una de vuelta al menu principal
  - Le da al cliente la posibilidad de Crear, Remover y Editar Partidos de la base de datos.
- Consultas Dinamicas
  - Posee 5 opciones y una de vuelta al menu.
  - Funciona como un buscador para facilitar los datos filtrados al usuario.

# Ejecucion del sistema



# Resultados

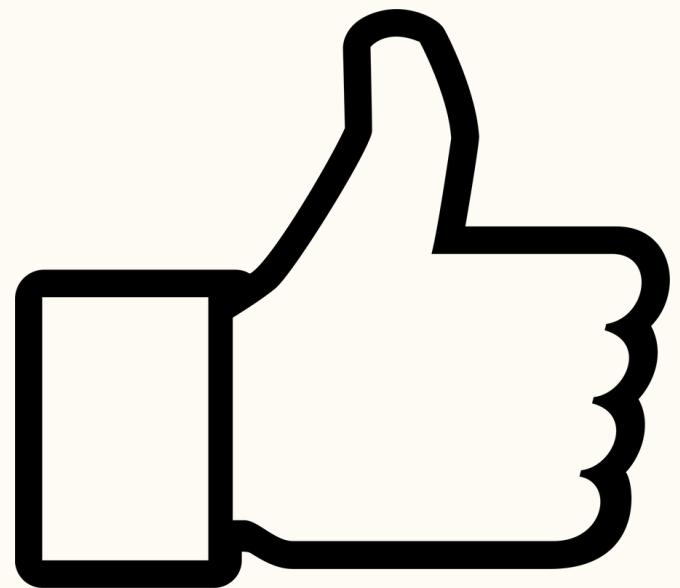
```
17 do {
18     cout << "\n--- Menú de Gestión de Datos ---\n";
19     cout << "1. Los primeros 5 partidos con mayor cantidad de goles\n";
20     cout << "2. Los goles totales a favor y en contra de cada equipo por competición\n";
21     cout << "3. Consultas Dinámicas\n";
22     cout << "4. Salir\n";
23 }
24
25 cout << "Por favor, seleccione una opción: ";
26
27 cout << "\nMenú Principal\n";
28 cout << "1 Estadísticas Curiosas\n";
29 cout << "2 Gestión de Partidos\n";
30 cout << "3 Consultas Dinámicas\n";
31 cout << "0 Salir\n";
32
33 cout << "Seleccione una opción: ";
34
35 cout << "Gracias por usar el programa. ¡Hasta luego! \ud83d\udcbb\n";
36 cout << "Tardo en segundos 0.086697\n";
37
38 mateopappalardo@MacBook-Air-de-Mateo output % 
```

Lín. 121, col. 1 Espacios:  
Compile Debug

## Observacion de la mejora en el tiempo de ejecucion y eficiencia

Nuestro producto tiene un tiempo de ejecucion entre consultas de 0.086 segundos. Reduce el tiempo de uso del sistema tanto del cliente como a nivel computacional.

# Resumen y conclusión



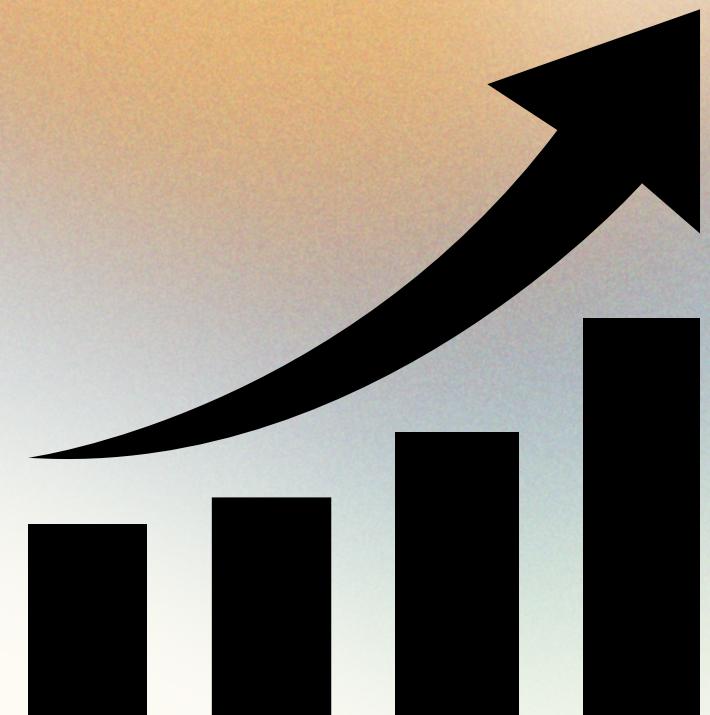
## Sistema Facil de Utilizar

Gracias a la implementacion de menus intuitivos y manual de usuario. Cualquiera puede usar nuestro producto



## Bajo Tiempo de ejecucion y compilado

Debido al correcto uso de estructuras y algortimo de datos. Logramos tener el menor tiempo en segundos de ejecucion



## Flexible y Escalable

Nuestro codigo se encuentra organizado y diagramado de forma tal que sea escalable a futuro