

**UNIVERSIDAD DE SONORA**

**Unidad Regional Centro**

**División de Ingeniería**

# **Ingeniería en Sistemas de Información**

**Base de Datos 1**

**Maestro: Navarro Hernández Rene Francisco**

**Grupo 1 – Hora 4 PM – 5 PM- Edif. 5G – L205**

**Nombres de Integrantes de Equipo:**

* **Vallejo Leyva Marcos**
* **Brianna Yaliczia Castillo Grijalva**
* **Daniela Garcia Lopez**

**Proyecto Final del Curso**

El proyecto es en equipo (3 integrantes) y van a desarrollar el proyecto que habían seleccionador anteriormente. Los elementos que se van a evaluar son los siguientes:

1. **Documento de diseño (20%)**
   1. Descripción general de los requerimientos para el Sistema de Gestión Deportiva

**1. Requerimientos Funcionales**

**1.1 Gestión de Jugadores**

* Agregar, modificar y eliminar jugadores.
* Visualizar la lista completa de jugadores y detalles específicos de cada uno, como nombre, género y dirección.
* Registrar y gestionar múltiples contactos o direcciones para cada jugador.

**1.2 Gestión de Juegos**

* Agregar, modificar y eliminar juegos.
* Mostrar detalles de cada juego, incluyendo su descripción.

**1.3 Gestión de Ligas**

* Crear, modificar y eliminar ligas.
* Asociar juegos a ligas específicas y gestionar estas asociaciones.

**1.4 Gestión de Equipos**

* Crear equipos indicando quién los ha creado.
* Modificar detalles de los equipos y disolver equipos existentes.
* Asociar jugadores a equipos y manejar su participación a lo largo del tiempo.

**1.5 Gestión de Rankings**

* Registrar y actualizar el ranking de los jugadores en diferentes juegos.
* Visualizar clasificaciones y estadísticas de rendimiento por juego.

**1.6 Gestión de Partidos**

* Programar y registrar resultados de partidos entre jugadores.
* Visualizar próximos partidos y resultados históricos.

**2. Requerimientos y Características Específicas**

**2.1 Tecnología**

* Java: Para el backend y la lógica de negocio.
* Swing: Para la interfaz gráfica.
* JDBC: Conexión y manipulación de la base de datos PostgreSQL.
* Java Transaction API (JTA): Para manejar transacciones a nivel de aplicación.

**2.2 Aplicación CRUD**

* Implementar funciones CRUD para cada entidad principal (Jugadores, Juegos, Ligas, Equipos, Partidos).
* Interfaz de usuario con formularios para la creación, visualización, actualización y eliminación de registros.
* Validación de datos antes de realizar operaciones en la base de datos.

**2.3 Procesos de Negocio**

* Gestión de Ligas y Torneos.
* Gestión de Equipos y Miembros.
* Ranking y Estadísticas de Jugadores.

**2.4 Manejo de Transacciones**

* Operaciones de múltiples acciones como una sola transacción para mantener laintegridad de la base de datos.
* Manejo de excepciones con bloques try-catch.

**2.5 Generación de Reportes**

* Reporte de Rankings de Jugadores.
* Reporte de Resultados de Partidos.
* Reporte de Actividad de Equipos.

**3. Requerimientos No Funcionales**

**3.1 Usabilidad**

* Interfaz de usuario intuitiva y fácil de usar.
* Accesibilidad desde dispositivos móviles y de escritorio.

**3.2 Rendimiento**

* Respuestas rápidas a consultas de la base de datos.
* Capacidad para manejar múltiples solicitudes simultáneas.

**3.3 Seguridad**

* Autenticación y autorización para proteger el acceso a la información.
* Seguridad en la transmisión de datos, idealmente mediante HTTPS.

**3.4 Escalabilidad**

* Capacidad para expandir y manejar un crecimiento en el número de usuarios y datos.

**3.5 Compatibilidad**

* Interoperabilidad con otros sistemas y plataformas.
* Soporte para múltiples navegadores y sistemas operativos.

**3.6 Mantenimiento y Soporte**

* Documentación completa y clara del sistema.
* Soporte técnico para resolver problemas y dudas de los usuarios.

**4. Consideraciones de Implementación**

* Tecnología de desarrollo: Stack tecnológico apropiado para el desarrollo del frontend y backend.
* Base de datos: PostgreSQL con herramientas ORM como SQLAlchemy o Django ORM.
* Servidores y despliegue: Configuración de servidor local o en la nube.

**5. Implementación**

**5.1 Interfaz de Usuario (Swing)**

* Crear ventanas y diálogos para cada una de las operaciones CRUD.
* Paneles separados para la gestión de ligas, juegos, jugadores, equipos y partidos.
* Menús para navegar entre las diferentes funcionalidades de la aplicación.

**5.2 Backend (Java + JDBC)**

* Clases en Java que representan las entidades de la base de datos.
* Utilizar JDBC para conectar con PostgreSQL y ejecutar consultas SQL.
* Clases de servicio para implementar la lógica de negocio.

**5.3 Reportes**

* Utilizar Swing para mostrar reportes en forma de tablas.
* Exportar reportes a formatos como PDF o Excel si se requiere.

**6. Pruebas y Despliegue**

**6.1 Pruebas**

* Pruebas unitarias y de integración.
* Pruebas de usabilidad.

**6.2 Despliegue**

* Configurar un entorno adecuado para la ejecución de la aplicación.

Esta descripción general proporciona una visión clara y organizada de los requerimientos, implementación y pruebas necesarias para desarrollar una aplicación robusta y funcional que cumpla con los requisitos especificados, utilizando Java, Swing y PostgreSQL.

* 1. **Esqueconceptual (ERD)**
  2. **Diagrama

     Descripción generada automáticamenteModelo**
  3. **Esquema relacional**
* **Players** (player\_id (PK), fist\_name, last\_name, gender, address)
* **Games** (game\_code (PK), game\_name, game\_description)
* **Leagues** (league\_id (PK), league\_name, league\_details)
* **Leagues\_Games** (leagues\_id (PF), game\_code (PF))
* **Players\_Game\_Ranking** (player\_id (PF), game\_code (PF), ranking)
* **Teams** (team\_id (PK), created\_by\_player\_id (FK), team\_name, date\_created, date\_disbanded)
* **Team\_Players** (team\_id (PF), player\_id (PF), date\_from (PK), date\_to)
* **Matches** (match\_id (PK), game\_code (FK), player\_1\_id (FK), player\_2\_id (FK), match\_date, result)
  1. **Diccionario de datos**

1. **Implementación de la base de datos (30%)**
   1. Implementación en PostgreSQL
   2. Implementación de restricciones de integridad referencial con llaves foráneas.
   3. Implementación de restricciones generales del dominio: Valores default + clausula CHECK
   4. Script para crear la base de datos con sus respectivas tablas

**Las restricciones también se encuentran en el siguiente script:**

* + 1. Creación de la Base de Datos:

CREATE DATABASE "Gaming\_Leagues"

WITH

OWNER = developer

TEMPLATE = template1

ENCODING = 'UTF8'

LOCALE\_PROVIDER = 'libc'

CONNECTION LIMIT = -1

IS\_TEMPLATE = False;

GRANT ALL ON DATABASE "Gaming\_Leagues" TO developer WITH GRANT OPTION;

* + 1. Creacion de Tablas.

-- Creación de tablas

BEGIN;

-- Jugadores

CREATE TABLE Players (

player\_id SERIAL PRIMARY KEY,

first\_name VARCHAR(100) NOT NULL,

last\_name VARCHAR(100) NOT NULL,

gender CHAR(1) CHECK (gender IN ('M', 'F', 'O')), -- M = Male, F = Female, O = Other

address VARCHAR(255)

);

-- Juegos

CREATE TABLE Games (

game\_code SERIAL PRIMARY KEY,

game\_name VARCHAR(100) NOT NULL,

game\_description TEXT

);

-- Ligas

CREATE TABLE Leagues (

league\_id SERIAL PRIMARY KEY,

league\_name VARCHAR(100) NOT NULL,

league\_details TEXT

);

-- Relación entre Ligas y Juegos

CREATE TABLE Leagues\_Games (

league\_id INTEGER NOT NULL,

game\_code INTEGER NOT NULL,

PRIMARY KEY (league\_id, game\_code),

FOREIGN KEY (league\_id) REFERENCES Leagues(league\_id) ON DELETE CASCADE,

FOREIGN KEY (game\_code) REFERENCES Games(game\_code) ON DELETE CASCADE

);

-- Ranking de Jugadores en Juegos

CREATE TABLE Players\_Game\_Ranking (

player\_id INTEGER NOT NULL,

game\_code INTEGER NOT NULL,

ranking INTEGER CHECK (ranking > 0),

PRIMARY KEY (player\_id, game\_code),

FOREIGN KEY (player\_id) REFERENCES Players(player\_id) ON DELETE CASCADE,

FOREIGN KEY (game\_code) REFERENCES Games(game\_code) ON DELETE CASCADE

);

-- Equipos

CREATE TABLE Teams (

team\_id SERIAL PRIMARY KEY,

created\_by\_player\_id INTEGER NOT NULL,

team\_name VARCHAR(100) NOT NULL,

date\_created DATE DEFAULT CURRENT\_DATE,

date\_disbanded DATE,

FOREIGN KEY (created\_by\_player\_id) REFERENCES Players(player\_id) ON DELETE SET NULL

);

-- Relación entre Equipos y Jugadores

CREATE TABLE Team\_Players (

team\_id INTEGER NOT NULL,

player\_id INTEGER NOT NULL,

date\_from DATE NOT NULL,

date\_to DATE,

PRIMARY KEY (team\_id, player\_id, date\_from),

FOREIGN KEY (team\_id) REFERENCES Teams(team\_id) ON DELETE CASCADE,

FOREIGN KEY (player\_id) REFERENCES Players(player\_id) ON DELETE CASCADE

);

-- Partidos

CREATE TABLE Matches (

match\_id SERIAL PRIMARY KEY,

game\_code INTEGER NOT NULL,

player\_1\_id INTEGER NOT NULL,

player\_2\_id INTEGER NOT NULL,

match\_date DATE DEFAULT CURRENT\_DATE,

result VARCHAR(10) CHECK (result IN ('win', 'lose', 'draw')),

FOREIGN KEY (game\_code) REFERENCES Games(game\_code),

FOREIGN KEY (player\_1\_id) REFERENCES Players(player\_id),

FOREIGN KEY (player\_2\_id) REFERENCES Players(player\_id)

);

COMMIT;

* 1. Script para insertar datos a todas las tablas de base de datos

BEGIN;

-- Insertar datos en Players

INSERT INTO Players (first\_name, last\_name, gender, address) VALUES

('John', 'Doe', 'M', '1234 Elm St'),

('Jane', 'Smith', 'F', '5678 Maple Ave');

-- Insertar datos en Games

INSERT INTO Games (game\_name, game\_description) VALUES

('Chess', 'Strategy board game'),

('Soccer', 'Team sport played with a spherical ball');

-- Insertar datos en Leagues

-- Insertar datos en Leagues

INSERT INTO Leagues (league\_name, league\_details) VALUES

('Junior League', 'For players under 20 years old'),

('Women''s League', 'All female players');

-- Insertar datos en Leagues\_Games

INSERT INTO Leagues\_Games (league\_id, game\_code) VALUES

(1, 1),

(2, 2);

-- Insertar datos en Players\_Game\_Ranking

INSERT INTO Players\_Game\_Ranking (player\_id, game\_code, ranking) VALUES

(1, 1, 1),

(2, 2, 2);

-- Insertar datos en Teams

INSERT INTO Teams (created\_by\_player\_id, team\_name) VALUES

(1, 'The Eagles'),

(2, 'The Hawks');

-- Insertar datos en Team\_Players

INSERT INTO Team\_Players (team\_id, player\_id, date\_from) VALUES

(1, 1, '2022-01-01'),

(2, 2, '2022-01-02');

-- Insertar datos en Matches

INSERT INTO Matches (game\_code, player\_1\_id, player\_2\_id, result) VALUES

(1, 1, 2, 'win'),

(2, 2, 1, 'lose');

COMMIT;

1. **Aplicación de escritorio (50%)**
   1. Implementación en Java + Swing
   2. Aplicación CRUD (create, read,  update, and delete)
   3. Implementar al menos tres procesos de negocio en la aplicación
   4. Usar manejo de transacciones
   5. Generar al menos 3 reportes con información relevante para la aplicación.