# Entrega 2 – Escuela de Jiu-Jitsu

## Descripción de la temática de la base de datos

La base de datos desarrollada tiene como objetivo gestionar de forma eficiente las operaciones internas de una escuela de Jiu-Jitsu. Abarca el registro de alumnos, asistencias a clases, cinturones, instructores, torneos, pagos e inscripciones. Está diseñada bajo el modelo relacional y busca representar las principales entidades y relaciones que se dan en el funcionamiento cotidiano de este tipo de institución.

## Introducción

Este proyecto propone el diseño de una base de datos relacional para una escuela de Jiu-Jitsu. El mismo busca organizar la información de manera estructurada y sin redundancia, permitiendo acceder fácilmente a los datos relevantes para la gestión de clases, alumnos, torneos y pagos.

## Objetivo

El objetivo es desarrollar un sistema de base de datos que permita a la escuela registrar y gestionar sus alumnos, instructores, clases, asistencias, inscripciones a torneos y pagos, con el fin de mejorar la eficiencia operativa y brindar una mejor trazabilidad de la información.

## Situación problemática

Actualmente, muchas escuelas de artes marciales gestionan sus operaciones mediante herramientas informales o manuales, como planillas de cálculo, anotaciones en papel o sistemas no integrados. Esto genera riesgos de pérdida de información, errores en los registros de asistencia o pagos y dificulta el seguimiento del progreso de los alumnos. Implementar una base de datos permite solucionar estas brechas.

## Modelo de negocio

La organización ficticia representada es una escuela de Jiu-Jitsu que ofrece clases grupales a diferentes niveles. Cada alumno se inscribe y abona una mensualidad. Se registra su asistencia, su cinturón actual (nivel), y puede participar en torneos. La escuela cuenta con varios instructores que dictan clases según horarios establecidos. El sistema está pensado para que la administración de la escuela pueda gestionar toda esta información de manera simple y centralizada.

## Listado de Tablas

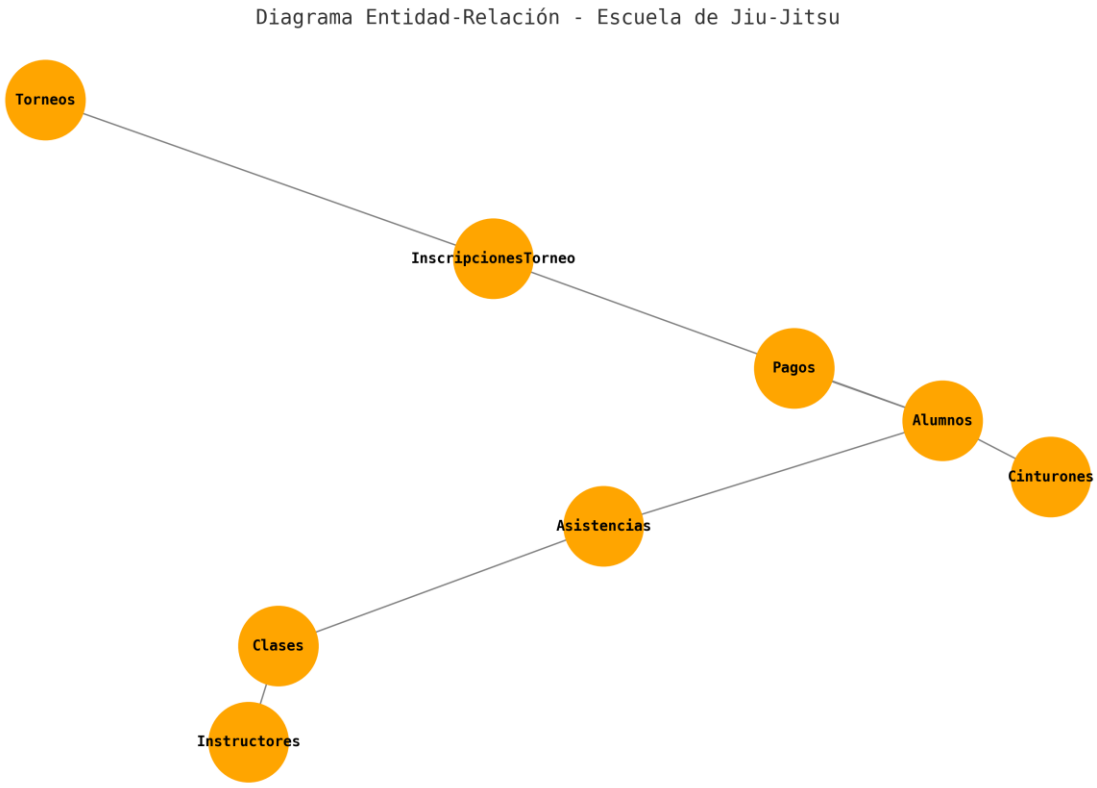
A continuación se presenta el listado de las tablas que conforman la base de datos de la escuela de Jiu-Jitsu. Se incluyen los nombres de las tablas, su descripción, los campos con su nombre abreviado y completo, tipo de dato y tipo de clave si corresponde.

| Tabla | Descripción | Campo (abreviado) | Campo (completo) | Tipo de dato | Clave |
|---|---|---|---|---|---|
| Alumnos | Información personal de los alumnos de la escuela | id_alumno | ID del alumno | INT | PK |
| Alumnos | | nombre | Nombre del alumno | VARCHAR(100) | |
| Alumnos | | apellido | Apellido del alumno | VARCHAR(100) | |
| Alumnos | | fecha_nacimiento | Fecha de nacimiento | DATE | |
| Alumnos | | cinturon_id | ID del cinturón actual | INT | FK |
| Cinturones | Cinturones disponibles según el nivel del alumno | id_cinturon | ID del cinturón | INT | PK |
| Cinturones | | color | Color del cinturón | VARCHAR(50) | |
| Cinturones | | nivel | Nivel asociado | INT | |
| Instructores | Instructores que dictan las clases | id_instructor | ID del instructor | INT | PK |
| Instructores | | nombre | Nombre del instructor | VARCHAR(100) | |
| Instructores | | apellido | Apellido del instructor | VARCHAR(100) | |
| Clases | Clases dictadas en la escuela | id_clase | ID de la clase | INT | PK |
| Clases | | fecha | Fecha de la clase | DATE | |
| Clases | | hora | Hora de la clase | TIME | |
| Clases | | id_instructor | ID del instructor | INT | FK |
| Asistencias | Registro de asistencias de los alumnos a las clases | id_asistencia | ID del registro de asistencia | INT | PK |
| Asistencias | | id_alumno | ID del alumno | INT | FK |
| Asistencias | | id_clase | ID de la clase | INT | FK |
| Asistencias | | presente | Asistencia | BOOLEAN | |

| | | | | | |
|---|---|---|---|---|---|
| | | | registrada | | |
| Torneos | Torneos organizados o en los que participa la escuela | id_torneo | ID del torneo | INT | PK |
| Torneos | | nombre | Nombre del torneo | VARCHAR(100) | |
| Torneos | | fecha | Fecha del torneo | DATE | |
| Inscripciones Torneo | Inscripción de alumnos a torneos | id_inscripcion | ID de la inscripción | INT | PK |
| Inscripciones Torneo | | id_torneo | ID del torneo | INT | FK |
| Inscripciones Torneo | | id_alumno | ID del alumno | INT | FK |
| Pagos | Pagos realizados por los alumnos | id_pago | ID del pago | INT | PK |
| Pagos | | id_alumno | ID del alumno | INT | FK |
| Pagos | | monto | Monto del pago | DECIMAL(10,2) | |
| Pagos | | fecha_pago | Fecha del pago | DATE | |

## Diagrama Entidad-Relación

A continuación se presenta el diagrama Entidad-Relación del proyecto. Este representa las entidades principales de la base de datos de la escuela de Jiu-Jitsu y las relaciones entre ellas, según el modelo relacional utilizado.



# Vistas

## vista_alumnos_cinturon

Descripción: Muestra los datos básicos de los alumnos junto con su cinturón actual.

Tablas utilizadas: Alumnos, Cinturones

Código SQL:

```
CREATE VIEW vista_alumnos_cinturon AS
SELECT
    a.id_alumno,
    a.nombre,
    a.apellido,
    c.color AS cinturon,
    c.nivel
FROM Alumnos a
JOIN Cinturones c ON a.cinturon_id = c.id_cinturon;
```

## vista_asistencias_por_clase

Descripción: Visualiza cuántos alumnos asistieron a cada clase, agrupado por fecha y hora.

Tablas utilizadas: Asistencias, Clases

Código SQL:

```
CREATE VIEW vista_asistencias_por_clase AS
SELECT
    c.id_clase,
    c.fecha,
    c.hora,
    COUNT(a.id_asistencia) AS total_asistentes
FROM Clases c
LEFT JOIN Asistencias a ON c.id_clase = a.id_clase AND a.presente = TRUE
GROUP BY c.id_clase, c.fecha, c.hora;
```

## vista_pagos_alumnos

Descripción: Muestra los pagos realizados por los alumnos con sus datos personales.

Tablas utilizadas: Pagos, Alumnos

Código SQL:

```
CREATE VIEW vista_pagos_alumnos AS
SELECT
```

```
    p.id_pago,
    a.nombre,
    a.apellido,
    p.monto,
    p.fecha_pago
FROM Pagos p
JOIN Alumnos a ON p.id_alumno = a.id_alumno;
```

## vista_torneos_por_alumno

Descripción: Muestra qué torneos disputó cada alumno.

Tablas utilizadas: InscripcionesTorneo, Torneos, Alumnos

Código SQL:

```
CREATE VIEW vista_torneos_por_alumno AS
SELECT
    a.nombre,
    a.apellido,
    t.nombre AS torneo,
    t.fecha
FROM InscripcionesTorneo i
JOIN Alumnos a ON i.id_alumno = a.id_alumno
JOIN Torneos t ON i.id_torneo = t.id_torneo;
```

# Funciones Personalizadas

## cantidad_torneos_alumno

Descripción: Devuelve la cantidad de torneos en los que ha participado un alumno según su ID.

Tablas utilizadas: InscripcionesTorneo

Código SQL:

```
DELIMITER $$

CREATE FUNCTION cantidad_torneos_alumno(id INT)
RETURNS INT
DETERMINISTIC
READS SQL DATA
BEGIN
    DECLARE total INT;

    SELECT COUNT(*) INTO total
    FROM InscripcionesTorneo
    WHERE id_alumno = id;

    RETURN total;
END $$

DELIMITER ;
```

## total_pagado_alumno

Descripción: Devuelve el total en pesos que ha abonado un alumno según su ID.

Tablas utilizadas: Pagos

Código SQL:

```
DELIMITER $$

CREATE FUNCTION total_pagado_alumno(id INT)
RETURNS DECIMAL(10,2)
DETERMINISTIC
READS SQL DATA
BEGIN
    DECLARE total DECIMAL(10,2);

    SELECT SUM(monto) INTO total
    FROM Pagos
    WHERE id_alumno = id;
```

```
    RETURN IFNULL(total, 0.00);
END $$

DELIMITER ;
```

# Procedimientos Almacenados (Stored Procedures)

## registrar_pago

Descripción: Registra un nuevo pago realizado por un alumno.

Tablas utilizadas: Pagos

Código SQL:

```
DELIMITER $$

CREATE PROCEDURE registrar_pago (
    IN alumno_id INT,
    IN monto_pago DECIMAL(10,2),
    IN fecha_pago DATE
)
BEGIN
    INSERT INTO Pagos (id_alumno, monto, fecha_pago)
    VALUES (alumno_id, monto_pago, fecha_pago);
END $$

DELIMITER ;
```

## inscribir_en_torneo

Descripción: Inscribe a un alumno en un torneo determinado.

Tablas utilizadas: InscripcionesTorneo

Código SQL:

```
DELIMITER $$

CREATE PROCEDURE inscribir_en_torneo (
    IN alumno_id INT,
    IN torneo_id INT
)
BEGIN
    INSERT INTO InscripcionesTorneo (id_alumno, id_torneo)
    VALUES (alumno_id, torneo_id);
END $$

DELIMITER ;
```

# Triggers

## log_asistencia

Descripción: Cada vez que se registra una nueva asistencia, se inserta automáticamente un registro en la tabla LogAsistencias, guardando la fecha, el ID del alumno y el ID de la clase. Esto permite auditar los ingresos de asistencia.

Tablas utilizadas: Asistencias, LogAsistencias

Código SQL:

```
CREATE TABLE LogAsistencias (
    id_log INT AUTO_INCREMENT PRIMARY KEY,
    id_alumno INT,
    id_clase INT,
    fecha_registro DATETIME
);

DELIMITER $$

CREATE TRIGGER log_asistencia
AFTER INSERT ON Asistencias
FOR EACH ROW
BEGIN
    INSERT INTO LogAsistencias (id_alumno, id_clase, fecha_registro)
    VALUES (NEW.id_alumno, NEW.id_clase, NOW());
END $$

DELIMITER ;
```