FORMATO DE TRABAJO FINAL

I. PORTADA

UNIVERSIDAD TÉCNICA DE AMBATO
Facultad de Ingeniería en Sistemas, Electrónica e Industrial
"Proyecto Académico de Fin de Semestre: marzo – julio 2025"

Título:	Cristian Ernesto Jurado Jacome
Carrera:	Software
Unidad de Organización Curricular:	Investigación
Línea de Investigación:	Profesional
Nivel y Paralelo:	Cuarto A
Alumnos:	Cristian Ernesto Jurado Jacome
Módulo y Docente:	Ing. Buenaño

II. INFORME DEL PROYECTO

2.1 Título

Control de Planificación y Ejecución de Partidos en el Mundial de Fútbol.

2.2 Objetivos

Mediante el desarrollo de este proyecto se pretenden lograr los siguientes objetivos

- Desarrollar una base de datos para la gestión integral de partidos de fútbol en un Mundial.
- Implementar un sistema de control para registrar resultados, goles, amonestaciones, expulsiones, sustituciones, autogoles, goles anulados, y mantener la tabla de posiciones.
- Diseñar triggers y procedimientos almacenados para asegurar la integridad y automatización de los datos relacionados con los partidos.

2.3 Palabras clave: (Palabra1, palabra2, palabra3…..)

- Base de Datos, Mundial de Fútbol, Gestión de Partidos, Triggers, Procedimientos Almacenados.

2.4 Resumen

Este proyecto tiene como objetivo principal la creación de una base de datos robusta para la planificación y ejecución de partidos en un Mundial de Fútbol. Se busca llevar un control detallado de los eventos ocurridos durante los encuentros, incluyendo resultados, goles (propios y anulados), sanciones (amonestaciones y expulsiones), y sustituciones de jugadores. La base de datos incluirá tablas para Selecciones, Jugadores, Árbitros, Estadios, Partidos, Asignación de Árbitros por Partido, Alineaciones, Goles, Amonestaciones, Sustituciones y una Tabla de Posiciones. Además, se implementarán una serie de triggers para asegurar la integridad de los datos, controlando aspectos como el número de titulares por equipo, la participación de jugadores y árbitros, y las reglas de juego (ej. inhabilitación por expulsión, límite de sustituciones). También se desarrollarán procedimientos almacenados para iniciar, anular goles y finalizar partidos, actualizando la tabla de posiciones automáticamente.

2.5 Introducción

- Este proyecto tiene como objetivo principal la creación de una base de datos robusta para la planificación y ejecución de partidos en un Mundial de Fútbol. Se busca llevar un control detallado de los eventos ocurridos durante los encuentros, incluyendo resultados, goles (propios y anulados), sanciones (amonestaciones y expulsiones), y sustituciones de jugadores. La base de datos incluirá tablas para Selecciones, Jugadores, Árbitros, Estadios, Partidos, Asignación de Árbitros por Partido, Alineaciones, Goles, Amonestaciones, Sustituciones y una Tabla de Posiciones. Además, se implementarán una serie de triggers para asegurar la integridad de los datos, controlando aspectos como el número de titulares por equipo, la participación de jugadores y árbitros, y las reglas de juego (ej. inhabilitación por expulsión, límite de sustituciones). También se desarrollarán procedimientos almacenados para iniciar, anular goles y finalizar partidos, actualizando la tabla de posiciones automáticamente.

2.6 Materiales y Metodología

2.6.1. Materiales:

- Oracle.

2.6.2 Metodología:

Para la conceptualización de la base de datos, se consideraron los principios de diseño relacional, asegurando la normalización para evitar redundancias y garantizar la integridad de los datos. Se identificaron las entidades clave involucradas en un Mundial de Fútbol:

- **Selecciones**: Identificadas por un ID, nombre, ranking y ubicación geográfica (ej. ARG Argentina 1 SUDAMERICA, ECU Ecuador 42 SUDAMERICA).
- **Jugadores**: Con ID, nombre, apellido, número de camiseta y la selección a la que pertenecen (ej. ARG01 EMILIANO MARTINEZ 1 ARG, ECU09 ENNER VALENCIA 13 ECU).
- **Árbitros**: Con ID, nombre, apellido, tipo de árbitro y nacionalidad (ej. ARB01 LUIS PEREZ FIFA1 URU).
- **Estadios**: Con ID, nombre, ciudad y capacidad (ej. AZ01 AZTECA MEXICO DF 100.000).
- **Partidos**: Incluyen número de partido, estadio, fecha y hora, selecciones participantes, estado del partido y marcadores (ej. 1, AZ01, 25/10/2026 16:00:00, ARG, ECU, P, NULL, NULL).

Recolección y Elaboración de Datos:

Se definieron las tablas y sus atributos, así como los tipos de datos y las claves primarias y foráneas para establecer las relaciones entre ellas. Se consideraron los siguientes detalles para el control de los partidos:

- **Resultados de los partidos:** Control de goles, expulsados, amonestados, sustituciones, autogoles, goles anulados, titulares, suplentes y la tabla de posiciones.
- **Relación Árbitros por Partido:** Se registra qué árbitros participan en cada partido y su rol (ej. ARB01 CENTRAL 1).
- **Alineaciones:** Se especifica el rol del jugador (Titular 'T' o Suplente 'S') en un partido (ej. ARG01 T 1, ARG10 S 2).
- **Goles:** Se registran con ID, minuto, descripción (ej. PENAL, AUTOGOL), ID del jugador que anota y el ID del partido (ej. 2432, 15, PENAL, ARG10, 1).
- **Amonestaciones:** Incluyen ID, minuto, tipo (ej. TA - Tarjeta Amarilla, TR - Tarjeta Roja), motivo y el jugador amonestado (ej. 2434, 25, TA, JUEGO BRUSCO, ARG10, 1).

- **Sustituciones:** Se registra el ID de la sustitución, minuto, motivo, el jugador que sale y el jugador que entra.
- **Tabla de Posiciones:** Con ID del equipo, partidos jugados (PJ), partidos ganados (PG), partidos empatados (PE), partidos perdidos (PP), goles a favor (GF), goles en contra (GC), diferencia de goles (GD) y puntos (PTOS).

Materiales y Métodos Utilizados (Procedimiento):

Se propone la implementación de la base de datos en un sistema de gestión de bases de datos relacional (SGBDR). Se definirán las siguientes lógicas de negocio a través de triggers y procedimientos almacenados:

Triggers:

- **Control de Titulares por Equipo:** Un trigger para asegurar que cada equipo tenga exactamente 11 titulares por partido.
- **Control de Doble Rol de Jugador:** Un trigger que impida que el mismo jugador sea titular más de una vez en el mismo partido.
- **Control de Titular y Suplente Simultáneo:** Un trigger que evite que un jugador sea titular y suplente en el mismo partido.
- **Control de Arbitraje Múltiple por Partido:** Un trigger para asegurar que un árbitro no esté más de una vez en un mismo partido.
- **Control de Nacionalidad de Árbitros:** Un trigger para evitar que un árbitro tenga la nacionalidad de los equipos que están jugando el partido.
- **Actualización de Goles al Marcador:** Un trigger que, después de insertar un gol, sume el gol a la selección correspondiente. Si la descripción es "AUTOGOL", el gol se sumará a la selección contraria. Se validará que el jugador que anota el gol esté en cancha.
- **Inhabilitación por Expulsión:** Un trigger para que un jugador expulsado quede inhabilitado de jugar el siguiente partido.
- **Restricción de Goles Post-Expulsión:** Un trigger que impida que un jugador expulsado haga goles.
- **Control de Doble Expulsión:** Un trigger para evitar que un jugador sea expulsado dos veces en el mismo partido.
- **Validación de Sustituciones:** Un trigger para controlar que el jugador que sale en una sustitución haya estado en cancha y que el jugador que entra esté en la lista de suplentes.
- **Límite de Sustituciones:** Un trigger que permita un máximo de 5 cambios por equipo en cada partido.
- **Control de Goles Post-Sustitución:** Un trigger que impida que un jugador que salió haga goles desde el minuto de su salida, y que un jugador que entró haya hecho goles antes de su entrada.

Procedimientos Almacenados:

- **INICIAR_PARTIDO(NUM_PAR):** Un procedimiento para cambiar el estado de un partido a 'J' (Jugando) y establecer el marcador en 0 a 0.
- **ANULAR_GOL(NUM_GOL):** Un procedimiento para revertir el estado de las tablas a la situación previa a la inserción de un gol (en caso de anulación por VAR).
- **FINALIZAR_PARTIDO(NUM_PAR):** Un procedimiento para cambiar el estado de un partido a 'F' (Finalizado) y sumar los puntos correspondientes a cada equipo en la tabla de posiciones.

2.7 Resultados y Discusión

CREACION DE LA BASE DE DATOS

```
PS C:\Users\LENOVO> sqlplus / as sysdba

SQL*Plus: Release 10.2.0.1.0 - Production on Dom Jul 6 22:21:31 2025

Copyright (c) 1982, 2005, Oracle.  All rights reserved.


Conectado a:
Oracle Database 10g Express Edition Release 10.2.0.1.0 - Production

SQL> CREATE USER MUNDIAL IDENTIFIED BY MUNDIAL123;

Usuario creado.

SQL> GRANT CONNECT, RESOURCE, UNLIMITED TABLESPACE TO MUNDIAL;

Concesi%n terminada correctamente.

SQL> CONNECT MUNDIAL/MUNDIAL123
Conectado.
SQL>
```

CREACION DE LAS TABLAS

TABLA →SELECCIONES

```
SQL> CREATE TABLE SELECCIONES (
  2    ID_SEL VARCHAR2(4) PRIMARY KEY,
  3    NOM_SEL VARCHAR2(20) NOT NULL,
  4    RANKING NUMBER NOT NULL,
  5    UBI_GEOG VARCHAR2(20) NOT NULL
  6  );

Tabla creada.
```

TABLA →JUGADORES

```
SQL>
SQL> CREATE TABLE JUGADORES (
  2    ID_JUG VARCHAR2(5) PRIMARY KEY,
  3    NOM_JUG VARCHAR2(12) NOT NULL,
  4    APE_JUG VARCHAR2(12) NOT NULL,
  5    NUM_CAM NUMBER NOT NULL,
  6    ID_SEL_PER VARCHAR2(4) NOT NULL REFERENCES SELECCIONES(ID_SEL)
  7  );

Tabla creada.
```

TABLA → ARBITROS

```
SQL> CREATE TABLE ARBITROS (
  2    ID_ARB VARCHAR2(5) PRIMARY KEY,
  3    NOM_ARB VARCHAR2(12) NOT NULL,
  4    APE_ARB VARCHAR2(12) NOT NULL,
  5    TIP_ARB VARCHAR2(20) NOT NULL,
  6    NAC_ARB VARCHAR2(20) NOT NULL
  7  );

Tabla creada.
```

TABLA → ESTADIOS

```
SQL> CREATE TABLE ESTADIOS (
  2    ID_EST VARCHAR2(4) PRIMARY KEY,
  3    NOM_EST VARCHAR2(20) NOT NULL,
  4    CIU_UBI VARCHAR2(20) NOT NULL,
  5    CAP_EST NUMBER NOT NULL
  6  );

Tabla creada.
```

INGRESO DE DATOS DE PRUEBA

DATOS PARA LA TABLA SELECCIONES

```
SQL> INSERT INTO SELECCIONES VALUES ('ARG', 'ARGENTINA', 1, 'SUDAMERICA');

1 fila creada.

SQL> INSERT INTO SELECCIONES VALUES ('ECU', 'ECUADOR', 42, 'SUDAMERICA');

1 fila creada.
```

DATOS PARA LA TABLA JUGADORES

```
SQL> INSERT INTO JUGADORES VALUES ('ARG01', 'LAUTARO', 'MARTINEZ', 1, 'ARG');

1 fila creada.

SQL> INSERT INTO JUGADORES VALUES ('ARG10', 'LIONEL', 'MESSI', 10, 'ARG');

1 fila creada.

SQL> INSERT INTO JUGADORES VALUES ('ECU15', 'ENNER', 'VALENCIA', 10, 'ECU');

1 fila creada.
```

CONSULTAS BASICAS

```
SQL> SELECT * FROM SELECCIONES;

ID_S NOM_SEL                RANKING UBI_GEOG
---- -------------------- ---------- --------------------
ARG  ARGENTINA                    1 SUDAMERICA
ECU  ECUADOR                     42 SUDAMERICA

SQL> SELECT * FROM JUGADORES;

ID_JU NOM_JUG      APE_JUG         NUM_CAM ID_S
----- ------------ ------------ ---------- ----
ARG01 LAUTARO      MARTINEZ              1 ARG
ARG10 LIONEL       MESSI                10 ARG
ECU15 ENNER        VALENCIA             10 ECU
```

Trigger: **Solo 11 titulares por equipo por partido**

```
SQL> CREATE TABLE PARTIDOS (
  2    NUM_PAR NUMBER PRIMARY KEY,
  3    EST_PAR VARCHAR2(4) NOT NULL,
  4    FEC_HOR_PAR DATE NOT NULL,
  5    SEL1 VARCHAR2(4) NOT NULL REFERENCES SELECCIONES(ID_SEL),
  6    SEL2 VARCHAR2(4) NOT NULL REFERENCES SELECCIONES(ID_SEL),
  7    ESTADO_PAR VARCHAR2(1) NOT NULL,
  8    NUM_GOL_SEL1 NUMBER,
  9    NUM_GOL_SEL2 NUMBER
 10  );

Tabla creada.

SQL> CREATE TABLE ALINEACIONES (
  2    ID_JUG_ALI VARCHAR2(5) REFERENCES JUGADORES(ID_JUG),
  3    ROL VARCHAR2(1), -- T = Titular, S = Suplente
  4    ID_PAR NUMBER REFERENCES PARTIDOS(NUM_PAR)
  5  );

Tabla creada.
```

```
SQL> CREATE OR REPLACE TRIGGER TRG_CONTROL_TITULARES
  2  BEFORE INSERT OR UPDATE OF ROL ON ALINEACIONES
  3  FOR EACH ROW
  4  DECLARE
  5    SELECCION_JUGADOR VARCHAR2(4);
  6    CANT_TITULARES NUMBER := 0;
  7  BEGIN
  8    -- Solo aplicar si el rol es Titular
  9    IF :NEW.ROL = 'T' THEN
 10
 11      -- Obtener la selección del jugador
 12      SELECT ID_SEL_PER INTO SELECCION_JUGADOR
 13      FROM JUGADORES
 14      WHERE ID_JUG = :NEW.ID_JUG_ALI;
 15
 16      -- Contar cuántos titulares ya tiene esa selección en ese partido
 17      SELECT COUNT(*) INTO CANT_TITULARES
 18      FROM ALINEACIONES A
 19      JOIN JUGADORES J ON A.ID_JUG_ALI = J.ID_JUG
 20      WHERE A.ROL = 'T'
 21        AND A.ID_PAR = :NEW.ID_PAR
 22        AND J.ID_SEL_PER = SELECCION_JUGADOR;
 23
 24      -- Validar que no pasen de 11
 25      IF CANT_TITULARES >= 11 THEN
 26        RAISE_APPLICATION_ERROR(-20001, 'Este equipo ya tiene 11 titulares en este partido.');
 27      END IF;
 28
 29    END IF;
 30  END;
 31  /

Disparador creado.
```

PROBAR EL TRIGGER

Insertar 11 titulares por ARG:

VALIDO

```
SQL> BEGIN
  2    FOR i IN 1..11 LOOP
  3      INSERT INTO ALINEACIONES VALUES ('ARG' || LPAD(i, 2, '0'), 'T', 1);
  4    END LOOP;
  5  END;
  6  /

Procedimiento PL/SQL terminado correctamente.
```

FALLA(INSERTAR JUGADOR 12)

```
SQL> INSERT INTO ALINEACIONES VALUES ('ARG12', 'T', 1);
INSERT INTO ALINEACIONES VALUES ('ARG12', 'T', 1)
            *
ERROR en lÝnea 1:
ORA-20001: Este equipo ya tiene 11 titulares en este partido.
```

VERIFICACION

```
SQL> SELECT A.ID_JUG_ALI
  2  FROM ALINEACIONES A
  3  JOIN JUGADORES J ON A.ID_JUG_ALI = J.ID_JUG
  4  WHERE A.ROL = 'T'
  5    AND A.ID_PAR = 1
  6    AND J.ID_SEL_PER = 'ARG';

ID_JU
-----
ARG01
ARG02
ARG03
ARG04
ARG05
ARG06
ARG07
ARG08
ARG09
ARG10
ARG11

11 filas seleccionadas.
```

TRIGGER: EVITAR QUE **EL MISMO JUGADOR** SE REGISTRE **MÁS DE UNA VEZ COMO TITULAR**

CREACION DEL TRIGGER

```
SQL> CREATE OR REPLACE TRIGGER TRG_UNICO_TITULAR
  2  BEFORE INSERT OR UPDATE OF ROL ON ALINEACIONES
  3  FOR EACH ROW
  4  DECLARE
  5    CANT_TITULARES NUMBER := 0;
  6  BEGIN
  7    IF :NEW.ROL = 'T' THEN
  8      SELECT COUNT(*) INTO CANT_TITULARES
  9      FROM ALINEACIONES
 10      WHERE ID_JUG_ALI = :NEW.ID_JUG_ALI
 11        AND ID_PAR = :NEW.ID_PAR
 12        AND ROL = 'T';
 13
 14      IF CANT_TITULARES > 0 THEN
 15        RAISE_APPLICATION_ERROR(-20002, 'El jugador ya es titular en este partido');
 16      END IF;
 17    END IF;
 18  END;
 19  /

Disparador creado.
```

Verifica que el jugador ya esté como titular

```
SQL> SELECT * FROM ALINEACIONES WHERE ID_JUG_ALI = 'ARG01' AND ID_PAR = 1;

ID_JU R     ID_PAR
----- - ----------
ARG01 T          1
```

VALICION EXITOSA (JUGADOR NO ES TITULAR DOS VECES EN UN MISMO PARTIDO)

```
SQL> INSERT INTO ALINEACIONES VALUES ('ARG01', 'T', 1);
INSERT INTO ALINEACIONES VALUES ('ARG01', 'T', 1)
            *
ERROR en lÝnea 1:
ORA-20002: El jugador ya es titular en este partido
```

TRIGGER : EL MISMO JUGADOR NO PUEDE SER **TITULAR Y SUPLENTE** EN EL MISMO PARTIDO
CREACION DEL TRIGGGER

```
SQL> CREATE OR REPLACE TRIGGER TRG_ROL_UNICO_POR_PARTIDO
  2  BEFORE INSERT ON ALINEACIONES
  3  FOR EACH ROW
  4  DECLARE
  5    EXISTE NUMBER := 0;
  6  BEGIN
  7    -- ¿Ya existe cualquier rol asignado para este jugador en este partido?
  8    SELECT COUNT(*) INTO EXISTE
  9    FROM ALINEACIONES
 10    WHERE ID_JUG_ALI = :NEW.ID_JUG_ALI
 11      AND ID_PAR = :NEW.ID_PAR;
 12
 13    IF EXISTE > 0 THEN
 14      RAISE_APPLICATION_ERROR(-20003, 'El jugador ya tiene un rol asignado en este partido (no puede ser titular y suplente a la vez)');
 15    END IF;
 16  END;
 17  /

Disparador creado.
```

Prueba del trigger (un jugador no puede ser suplente y titular en un mismo partido)

```
SQL> INSERT INTO ALINEACIONES VALUES ('ARG01', 'S', 1);
INSERT INTO ALINEACIONES VALUES ('ARG01', 'S', 1)
            *
ERROR en lÝnea 1:
ORA-20003: El jugador ya tiene un rol asignado en este partido (no puede ser
titular y suplente a la vez)
```

Prueba del trigger (un jugador es titular)

```
SQL> INSERT INTO ALINEACIONES VALUES ('ARG12', 'S', 1);

1 fila creada.
```

TRIGGER #4 UN ÁRBITRO NO PUEDE ESTAR MÁS DE UNA VEZ EN EL MISMO PARTIDO
TABLA → ARBITROXPARTIDO

```
SQL> CREATE TABLE ARBITROSXPARTIDO (
  2    ID_ARB_PAR VARCHAR2(5) REFERENCES ARBITROS(ID_ARB),
  3    ROL VARCHAR2(12) NOT NULL,
  4    ID_PAR NUMBER REFERENCES PARTIDOS(NUM_PAR)
  5  );

Tabla creada.
```

Creacion del trigger

```
SQL> CREATE OR REPLACE TRIGGER TRG_UNICO_ARB_POR_PARTIDO
  2  BEFORE INSERT ON ARBITROSXPARTIDO
  3  FOR EACH ROW
  4  DECLARE
  5    EXISTE NUMBER := 0;
  6  BEGIN
  7    SELECT COUNT(*) INTO EXISTE
  8    FROM ARBITROSXPARTIDO
  9    WHERE ID_ARB_PAR = :NEW.ID_ARB_PAR
 10      AND ID_PAR = :NEW.ID_PAR;
 11
 12    IF EXISTE > 0 THEN
 13      RAISE_APPLICATION_ERROR(-20004, 'El árbitro ya está asignado a este partido');
 14    END IF;
 15  END;
 16  /

Disparador creado.
```

Insertar datos de prueba

```
SQL> INSERT INTO ARBITROS VALUES ('ARB01', 'Luis', 'Pérez', 'FIFA', 'URU');

1 fila creada.

SQL> INSERT INTO ARBITROS VALUES ('ARB02', 'Marta', 'Suárez', 'FIFA', 'PER');

1 fila creada.

SQL> COMMIT;

Confirmaci¾n terminada.
```

Insertar el árbitro en el partido 1

```
SQL> INSERT INTO ARBITROSXPARTIDO VALUES ('ARB01', 'Principal', 1);

1 fila creada.
```

Insertar el mismo arbitro otra vez en el mismo partido

```
SQL> INSERT INTO ARBITROSXPARTIDO VALUES ('ARB01', 'Asistente', 1);
INSERT INTO ARBITROSXPARTIDO VALUES ('ARB01', 'Asistente', 1)
            *
ERROR en lÝnea 1:
ORA-20004: El árbitro ya está asignado a este partido
```

TRIGGER 5: EL ÁRBITRO NO DEBE TENER LA MISMA NACIONALIDAD QUE ALGUNO DE LOS EQUIPOS QUE JUEGAN EL PARTIDO

CREACION DEL TRIGGER

```
SQL> CREATE OR REPLACE TRIGGER TRG_NACIONALIDAD_ARB_PROHIBIDA
  2  BEFORE INSERT ON ARBITROSXPARTIDO
  3  FOR EACH ROW
  4  DECLARE
  5    NAC_ARB VARCHAR2(20);
  6    SEL1 VARCHAR2(4);
  7    SEL2 VARCHAR2(4);
  8  BEGIN
  9    -- Obtener nacionalidad del árbitro
 10    SELECT NAC_ARB INTO NAC_ARB
 11    FROM ARBITROS
 12    WHERE ID_ARB = :NEW.ID_ARB_PAR;
 13
 14    -- Obtener selecciones del partido
 15    SELECT SEL1, SEL2 INTO SEL1, SEL2
 16    FROM PARTIDOS
 17    WHERE NUM_PAR = :NEW.ID_PAR;
 18
 19    -- Verificar si la nacionalidad del árbitro coincide con algún equipo
 20    IF NAC_ARB = SEL1 OR NAC_ARB = SEL2 THEN
 21      RAISE_APPLICATION_ERROR(-20005, 'El árbitro no puede tener la nacionalidad de un equipo participante');
 22    END IF;
 23  END;
 24  /

Disparador creado.
```

Insertar árbitros con distintas nacionalidades

```
SQL> INSERT INTO ARBITROS VALUES ('ARB03', 'Carlos', 'Gómez', 'FIFA', 'ARG');

1 fila creada.

SQL> INSERT INTO ARBITROS VALUES ('ARB04', 'Ana', 'López', 'FIFA', 'MEX');

1 fila creada.

SQL> COMMIT;

Confirmaci¾n terminada.
```

Insertar árbitros con distintas nacionalidades

```
SQL> INSERT INTO ARBITROSXPARTIDO VALUES ('ARB03', 'Principal', 1);
INSERT INTO ARBITROSXPARTIDO VALUES ('ARB03', 'Principal', 1)
            *
ERROR en lÝnea 1:
ORA-20005: El árbitro no puede tener la nacionalidad de un equipo participante
```

PROCEDIMIENTO: INICIAR_PARTIDO (cambia estado y pone marcador en 0–0)

```
SQL> CREATE OR REPLACE PROCEDURE INICIAR_PARTIDO(P_NUM_PAR NUMBER)
  2  IS
  3  BEGIN
  4    UPDATE PARTIDOS
  5    SET ESTADO_PAR = 'J',
  6        NUM_GOL_SEL1 = 0,
  7        NUM_GOL_SEL2 = 0
  8    WHERE NUM_PAR = P_NUM_PAR;
  9
 10    IF SQL%ROWCOUNT = 0 THEN
 11      RAISE_APPLICATION_ERROR(-20006, 'El partido no existe');
 12    END IF;
 13  END;
 14  /

Procedimiento creado.
```

PRUEBA

```
SQL> SELECT NUM_PAR, ESTADO_PAR, NUM_GOL_SEL1, NUM_GOL_SEL2
  2  FROM PARTIDOS
  3  WHERE NUM_PAR = 1;

   NUM_PAR E NUM_GOL_SEL1 NUM_GOL_SEL2
---------- - ------------ ------------
         1 P            0            0
```

```
SQL> EXEC INICIAR_PARTIDO(1);

Procedimiento PL/SQL terminado correctamente.
```

```
SQL> SELECT NUM_PAR, ESTADO_PAR, NUM_GOL_SEL1, NUM_GOL_SEL2
  2  FROM PARTIDOS
  3  WHERE NUM_PAR = 1;

   NUM_PAR E NUM_GOL_SEL1 NUM_GOL_SEL2
---------- - ------------ ------------
         1 J            0            0
```

TRG_SUMAR_GOL: SUMA EL GOL AL MARCADOR DEL PARTIDO Y MANEJA AUTOGOLES

```
SQL> CREATE TABLE GOLES (
  2    ID_GOL NUMBER PRIMARY KEY,
  3    MIN_GOL NUMBER NOT NULL,
  4    DES_GOL VARCHAR2(30) NOT NULL,
  5    ID_JUG_GOL VARCHAR2(5) REFERENCES JUGADORES(ID_JUG),
  6    ID_PAR NUMBER REFERENCES PARTIDOS(NUM_PAR)
  7  );

Tabla creada.
```

CREACION DEL TRIGGER

```
SQL> CREATE OR REPLACE TRIGGER TRG_SUMAR_GOL
  2  AFTER INSERT ON GOLES
  3  FOR EACH ROW
  4  DECLARE
  5    SEL_JUGADOR VARCHAR2(4);
  6    SEL1 VARCHAR2(4);
  7    SEL2 VARCHAR2(4);
  8    ROL_JUG VARCHAR2(1);
  9  BEGIN
 10    -- Obtener selección del jugador
 11    SELECT ID_SEL_PER INTO SEL_JUGADOR
 12    FROM JUGADORES
 13    WHERE ID_JUG = :NEW.ID_JUG_GOL;
 14
 15    -- Obtener selecciones del partido
 16    SELECT SEL1, SEL2 INTO SEL1, SEL2
 17    FROM PARTIDOS
 18    WHERE NUM_PAR = :NEW.ID_PAR;
 19
 20    -- Verificar si el jugador está en cancha como titular
 21    SELECT ROL INTO ROL_JUG
 22    FROM ALINEACIONES
 23    WHERE ID_JUG_ALI = :NEW.ID_JUG_GOL
 24      AND ID_PAR = :NEW.ID_PAR;
 25
 26    IF ROL_JUG <> 'T' THEN
 27      RAISE_APPLICATION_ERROR(-20007, 'El jugador no está en cancha como titular');
 28    END IF;
 29
 30    -- Gol normal: suma a su equipo
 31    IF :NEW.DES_GOL <> 'AUTOGOL' THEN
 32      IF SEL_JUGADOR = SEL1 THEN
 33        UPDATE PARTIDOS SET NUM_GOL_SEL1 = NVL(NUM_GOL_SEL1, 0) + 1 WHERE NUM_PAR = :NEW.ID_PAR;
 34      ELSE
 35        UPDATE PARTIDOS SET NUM_GOL_SEL2 = NVL(NUM_GOL_SEL2, 0) + 1 WHERE NUM_PAR = :NEW.ID_PAR;
 36      END IF;
 37    ELSE
 38      -- AUTOGOL: suma al rival
 39      IF SEL_JUGADOR = SEL1 THEN
 40        UPDATE PARTIDOS SET NUM_GOL_SEL2 = NVL(NUM_GOL_SEL2, 0) + 1 WHERE NUM_PAR = :NEW.ID_PAR;
 41      ELSE
 42        UPDATE PARTIDOS SET NUM_GOL_SEL1 = NVL(NUM_GOL_SEL1, 0) + 1 WHERE NUM_PAR = :NEW.ID_PAR;
 43      END IF;
 44    END IF;
 45  END;
 46  /
```

Insertar un gol (caso exitoso)

```
SQL> INSERT INTO GOLES VALUES (1, 20, 'PENAL', 'ARG10', 1);

1 fila creada.
```

Insertar un autogol

```
SQL> INSERT INTO GOLES VALUES (2, 30, 'AUTOGOL', 'ARG10', 1);

1 fila creada.
```

EVIDENCIA DE QUE SE SUMARON LOS GOLES

```
SQL> SELECT NUM_PAR, NUM_GOL_SEL1, NUM_GOL_SEL2 FROM PARTIDOS WHERE NUM_PAR = 1;

   NUM_PAR NUM_GOL_SEL1 NUM_GOL_SEL2
---------- ------------ ------------
         1            1            1
```

TRIGEER: UN JUGADOR EXPULSADO QUEDA INHABILITADO: NO PUEDE JUGAR EL SIGUIENTE PARTIDO NI HACER GOLES

TABLA → AMONESTACIONES

```
SQL> CREATE TABLE AMONESTADOS (
  2    ID_AMO NUMBER PRIMARY KEY,
  3    MIN_AMO NUMBER NOT NULL,
  4    TIP_AMO VARCHAR2(2) NOT NULL, -- TA (amarilla), TR (roja)
  5    MOTIVO VARCHAR2(30),
  6    ID_JUG_AMO VARCHAR2(5) REFERENCES JUGADORES(ID_JUG),
  7    ID_PAR NUMBER REFERENCES PARTIDOS(NUM_PAR)
  8  );

Tabla creada.
```

Tabla auxiliar

```
SQL> CREATE TABLE JUGADORES_INHABILITADOS (
  2    ID_JUG VARCHAR2(5) PRIMARY KEY,
  3    RAZON VARCHAR2(50)
  4  );

Tabla creada.
```

Creacion del trigger

```
SQL> CREATE OR REPLACE TRIGGER TRG_EXPULSION_INHABILITA
  2  AFTER INSERT ON AMONESTADOS
  3  FOR EACH ROW
  4  BEGIN
  5    -- Si es tarjeta roja, agregar al jugador como inhabilitado
  6    IF :NEW.TIP_AMO = 'TR' THEN
  7      INSERT INTO JUGADORES_INHABILITADOS (ID_JUG, RAZON)
  8      VALUES (:NEW.ID_JUG_AMO, 'EXPULSADO - INHABILITADO');
  9    END IF;
 10  END;
 11  /

Disparador creado.
```

Crear el trigger que impide que hagan goles

```
SQL> CREATE OR REPLACE TRIGGER TRG_GOL_INHABILITADO
  2  BEFORE INSERT ON GOLES
  3  FOR EACH ROW
  4  DECLARE
  5    EXISTE NUMBER := 0;
  6  BEGIN
  7    SELECT COUNT(*) INTO EXISTE
  8    FROM JUGADORES_INHABILITADOS
  9    WHERE ID_JUG = :NEW.ID_JUG_GOL;
 10
 11    IF EXISTE > 0 THEN
 12      RAISE_APPLICATION_ERROR(-20008, 'Jugador expulsado: no puede hacer goles');
 13    END IF;
 14  END;
 15  /

Disparador creado.
```

Expulsar a ARG10 (Messi)

```
SQL> INSERT INTO AMONESTADOS VALUES (1, 35, 'TR', 'Juego brusco', 'ARG10', 1);

1 fila creada.
```

Ver si fue marcado como inhabilitado

```
SQL> SELECT * FROM JUGADORES_INHABILITADOS;

ID_JU RAZON
----- --------------------------------------------------
ARG10 EXPULSADO - INHABILITADO
```

Intentar que haga un gol (debe fallar)

```
SQL> INSERT INTO GOLES VALUES (3, 60, 'CABEZAZO', 'ARG10', 1);
INSERT INTO GOLES VALUES (3, 60, 'CABEZAZO', 'ARG10', 1)
            *
ERROR en lÝnea 1:
ORA-20008: Jugador expulsado: no puede hacer goles
```

TRIGGER: NO SE PUEDE EXPULSAR DOS VECES AL MISMO JUGADOR EN EL MISMO PARTIDO

```
SQL> CREATE OR REPLACE TRIGGER TRG_EXPULSION_UNICA
  2  BEFORE INSERT ON AMONESTADOS
  3  FOR EACH ROW
  4  DECLARE
  5    EXISTE NUMBER := 0;
  6  BEGIN
  7    -- Verificar si el jugador ya tiene tarjeta roja en ese partido
  8    SELECT COUNT(*) INTO EXISTE
  9    FROM AMONESTADOS
 10    WHERE ID_JUG_AMO = :NEW.ID_JUG_AMO
 11      AND ID_PAR = :NEW.ID_PAR
 12      AND TIP_AMO = 'TR';
 13
 14    IF EXISTE > 0 THEN
 15      RAISE_APPLICATION_ERROR(-20009, 'El jugador ya fue expulsado en este partido');
 16    END IF;
 17  END;
 18  /

Disparador creado.
```

Intentar una segunda expulsión (debe fallar)

```
SQL> INSERT INTO AMONESTADOS VALUES (2, 45, 'TR', 'Mano intencional', 'ARG10', 1);
INSERT INTO AMONESTADOS VALUES (2, 45, 'TR', 'Mano intencional', 'ARG10', 1)
            *
ERROR en lÝnea 1:
ORA-20009: El jugador ya fue expulsado en este partido
```

TRIGGER: QUE CONTROLA SUSTITUCIONES: SOLO PUEDE SALIR SI ESTUVO EN CANCHA, Y ENTRAR SI ERA SUPLENTE
TABLA → SUSTITUCIONES

```
SQL> CREATE TABLE SUSTITUCIONES (
  2    ID_SUS NUMBER PRIMARY KEY,
  3    MIN_SUS NUMBER NOT NULL,
  4    MOTIVO VARCHAR2(30),
  5    ID_JUG_SALE VARCHAR2(5) REFERENCES JUGADORES(ID_JUG),
  6    ID_JUG_ENTRA VARCHAR2(5) REFERENCES JUGADORES(ID_JUG),
  7    ID_PAR NUMBER REFERENCES PARTIDOS(NUM_PAR)
  8  );

Tabla creada.
```

Creacion del trigger

```
SQL> CREATE OR REPLACE TRIGGER TRG_CONTROL_SUSTITUCION_ROLES
  2  BEFORE INSERT ON SUSTITUCIONES
  3  FOR EACH ROW
  4  DECLARE
  5    ROL_SALE VARCHAR2(1);
  6    ROL_ENTRA VARCHAR2(1);
  7  BEGIN
  8    -- Verificar si el que sale estaba en cancha como titular
  9    SELECT ROL INTO ROL_SALE
 10    FROM ALINEACIONES
 11    WHERE ID_JUG_ALI = :NEW.ID_JUG_SALE
 12      AND ID_PAR = :NEW.ID_PAR;
 13
 14    -- Verificar si el que entra estaba como suplente
 15    SELECT ROL INTO ROL_ENTRA
 16    FROM ALINEACIONES
 17    WHERE ID_JUG_ALI = :NEW.ID_JUG_ENTRA
 18      AND ID_PAR = :NEW.ID_PAR;
 19
 20    IF ROL_SALE <> 'T' THEN
 21      RAISE_APPLICATION_ERROR(-20010, 'El jugador que sale no es titular');
 22    END IF;
 23
 24    IF ROL_ENTRA <> 'S' THEN
 25      RAISE_APPLICATION_ERROR(-20011, 'El jugador que entra no es suplente');
 26    END IF;
 27  END;
 28  /

Disparador creado.
```

Validación del trigger

```
SQL> INSERT INTO ALINEACIONES VALUES ('ARG12', 'S', 1);
INSERT INTO ALINEACIONES VALUES ('ARG12', 'S', 1)
            *
ERROR en lÝnea 1:
ORA-20003: El jugador ya tiene un rol asignado en este partido (no puede ser
titular y suplente a la vez)
```

Inserción válida (sale titular, entra suplente)

```
SQL> INSERT INTO SUSTITUCIONES VALUES (1, 60, 'Cambio táctico', 'ARG01', 'ARG12', 1);

1 fila creada.
```

Caso inválido 1: El que sale no era titular

```
SQL> INSERT INTO SUSTITUCIONES VALUES (2, 65, 'Lesión', 'ARG12', 'ARG10', 1);
INSERT INTO SUSTITUCIONES VALUES (2, 65, 'Lesión', 'ARG12', 'ARG10', 1)
            *
ERROR en lÝnea 1:
ORA-20010: El jugador que sale no es titular
```

TRIGGER: SE PERMITEN SOLO 5 CAMBIOS POR EQUIPO EN CADA PARTIDO

Creacion del trigger

```
SQL> CREATE OR REPLACE TRIGGER TRG_LIMITE_CAMBIOS
  2  BEFORE INSERT ON SUSTITUCIONES
  3  FOR EACH ROW
  4  DECLARE
  5    SEL_ENTRA VARCHAR2(4);
  6    TOTAL_CAMBIOS NUMBER := 0;
  7  BEGIN
  8    -- Obtener selección del jugador que entra
  9    SELECT ID_SEL_PER INTO SEL_ENTRA
 10    FROM JUGADORES
 11    WHERE ID_JUG = :NEW.ID_JUG_ENTRA;
 12
 13    -- Contar cuántas sustituciones lleva ese equipo en el partido
 14    SELECT COUNT(*) INTO TOTAL_CAMBIOS
 15    FROM SUSTITUCIONES S
 16    JOIN JUGADORES J ON S.ID_JUG_ENTRA = J.ID_JUG
 17    WHERE S.ID_PAR = :NEW.ID_PAR
 18      AND J.ID_SEL_PER = SEL_ENTRA;
 19
 20    IF TOTAL_CAMBIOS >= 5 THEN
 21      RAISE_APPLICATION_ERROR(-20012, 'No se permiten más de 5 sustituciones por equipo en un partido');
 22    END IF;
 23  END;
 24  /

Disparador creado.
```

```
SQL> BEGIN
  2    FOR i IN 13..18 LOOP
  3      INSERT INTO JUGADORES VALUES ('ARG' || LPAD(i, 2, '0'), 'Jugador' || i, 'Apellido' || i, i, 'ARG');
  4      INSERT INTO ALINEACIONES VALUES ('ARG' || LPAD(i, 2, '0'), 'S', 1);
  5    END LOOP;
  6  END;
  7  /

Procedimiento PL/SQL terminado correctamente.
```

```
ERROR en línea 1:
ORA-20012: No se permiten más de 5 sustituciones por equipo en un partido
```

TRIGGER: SI UN JUGADOR SALE, NO PUEDE HACER GOLES DESPUÉS. SI ENTRA, NO PUEDE HABER HECHO GOLES ANTES DEL MINUTO DE INGRESO.

Creacion del trigger

```
SQL> CREATE OR REPLACE TRIGGER TRG_TIEMPO_GOLES_SALIDA
  2  BEFORE INSERT ON GOLES
  3  FOR EACH ROW
  4  DECLARE
  5    MIN_SALIDA NUMBER;
  6  BEGIN
  7    SELECT MIN(MIN_SUS) INTO MIN_SALIDA
  8    FROM SUSTITUCIONES
  9    WHERE ID_JUG_SALE = :NEW.ID_JUG_GOL
 10      AND ID_PAR = :NEW.ID_PAR;
 11
 12    IF :NEW.MIN_GOL > MIN_SALIDA THEN
 13      RAISE_APPLICATION_ERROR(-20014, 'El jugador no puede hacer goles después de salir del campo');
 14    END IF;
 15
 16  EXCEPTION
 17    WHEN NO_DATA_FOUND THEN
 18      NULL;
 19    WHEN TOO_MANY_ROWS THEN
 20      NULL;
 21  END;
 22  /

Disparador creado.
```

Pruebas del trigger

```
SQL> INSERT INTO GOLES VALUES (100, 50, 'TIRO LIBRE', 'ARG12', 1);
INSERT INTO GOLES VALUES (100, 50, 'TIRO LIBRE', 'ARG12', 1)
            *
ERROR en lÝnea 1:
ORA-20013: El jugador no puede hacer goles antes de entrar al campo
```

PROCEDURE ANULAR_GOL

Creacion del procedure

```
SQL> CREATE OR REPLACE PROCEDURE ANULAR_GOL(P_ID_GOL NUMBER)
  2  IS
  3    V_DES_GOL GOLES.DES_GOL%TYPE;
  4    V_ID_JUG  GOLES.ID_JUG_GOL%TYPE;
  5    V_ID_PAR  GOLES.ID_PAR%TYPE;
  6    V_SEL_JUG VARCHAR2(4);
  7    V_SEL1    VARCHAR2(4);
  8    V_SEL2    VARCHAR2(4);
  9  BEGIN
 10    -- Obtener info del gol
 11    SELECT DES_GOL, ID_JUG_GOL, ID_PAR INTO V_DES_GOL, V_ID_JUG, V_ID_PAR
 12    FROM GOLES
 13    WHERE ID_GOL = P_ID_GOL;
 14
 15    -- Selección del jugador
 16    SELECT ID_SEL_PER INTO V_SEL_JUG
 17    FROM JUGADORES
 18    WHERE ID_JUG = V_ID_JUG;
 19
 20    -- Selecciones del partido
 21    SELECT SEL1, SEL2 INTO V_SEL1, V_SEL2
 22    FROM PARTIDOS
 23    WHERE NUM_PAR = V_ID_PAR;
 24
 25    -- Restar el gol al equipo correcto
 26    IF V_DES_GOL <> 'AUTOGOL' THEN
 27      IF V_SEL_JUG = V_SEL1 THEN
 28        UPDATE PARTIDOS SET NUM_GOL_SEL1 = NUM_GOL_SEL1 - 1 WHERE NUM_PAR = V_ID_PAR;
 29      ELSE
 30        UPDATE PARTIDOS SET NUM_GOL_SEL2 = NUM_GOL_SEL2 - 1 WHERE NUM_PAR = V_ID_PAR;
 31      END IF;
 32    ELSE
 33      -- AUTOGOL: se resta al equipo rival
 34      IF V_SEL_JUG = V_SEL1 THEN
 35        UPDATE PARTIDOS SET NUM_GOL_SEL2 = NUM_GOL_SEL2 - 1 WHERE NUM_PAR = V_ID_PAR;
 36      ELSE
 37        UPDATE PARTIDOS SET NUM_GOL_SEL1 = NUM_GOL_SEL1 - 1 WHERE NUM_PAR = V_ID_PAR;
 38      END IF;
 39    END IF;
 40
 41    -- Borrar el gol
 42    DELETE FROM GOLES WHERE ID_GOL = P_ID_GOL;
 43
 44  EXCEPTION
 45    WHEN NO_DATA_FOUND THEN
 46      RAISE_APPLICATION_ERROR(-20020, 'El ID de gol no existe');
 47  END;
 48  /

Procedimiento creado.
```

VALIDACION DEL PROCEDURE

```
SQL> INSERT INTO GOLES VALUES (201, 80, 'CABEZAZO', 'ARG08', 1);

1 fila creada.

SQL> EXEC ANULAR_GOL(201);

Procedimiento PL/SQL terminado correctamente.
```

PROCEDIMIENTO: FINALIZAR_PARTIDO QUE ACTUALIZA LA TABLA DE POSICIONES
TABLA → TABLA POSICIONES

```
SQL> CREATE TABLE TABLA (
  2     ID_EQU VARCHAR2(4) PRIMARY KEY REFERENCES SELECCIONES(ID_SEL),
  3     PJ NUMBER DEFAULT 0,
  4     PG NUMBER DEFAULT 0,
  5     PE NUMBER DEFAULT 0,
  6     PP NUMBER DEFAULT 0,
  7     GF NUMBER DEFAULT 0,
  8     GC NUMBER DEFAULT 0,
  9     GD NUMBER DEFAULT 0,
 10     PTOS NUMBER DEFAULT 0
 11   );

Tabla creada.
```

Creacion del procedimiento

```
SQL> CREATE OR REPLACE PROCEDURE FINALIZAR_PARTIDO(P_NUM_PAR NUMBER)
  2  IS
  3    V_SEL1 VARCHAR2(4);
  4    V_SEL2 VARCHAR2(4);
  5    G1 NUMBER;
  6    G2 NUMBER;
  7  BEGIN
  8    -- Obtener info del partido
  9    SELECT SEL1, SEL2, NUM_GOL_SEL1, NUM_GOL_SEL2
 10    INTO V_SEL1, V_SEL2, G1, G2
 11    FROM PARTIDOS
 12    WHERE NUM_PAR = P_NUM_PAR;
 13
 14    -- Marcar partido como finalizado
 15    UPDATE PARTIDOS
 16    SET ESTADO_PAR = 'F'
 17    WHERE NUM_PAR = P_NUM_PAR;
 18
 19    -- Equipo 1
 20    UPDATE TABLA
 21    SET PJ = PJ + 1,
 22        GF = GF + G1,
 23        GC = GC + G2,
 24        GD = GD + (G1 - G2),
 25        PG = PG + CASE WHEN G1 > G2 THEN 1 ELSE 0 END,
 26        PE = PE + CASE WHEN G1 = G2 THEN 1 ELSE 0 END,
 27        PP = PP + CASE WHEN G1 < G2 THEN 1 ELSE 0 END,
 28        PTOS = PTOS + CASE WHEN G1 > G2 THEN 3 WHEN G1 = G2 THEN 1 ELSE 0 END
 29    WHERE ID_EQU = V_SEL1;
 30
 31    -- Equipo 2
 32    UPDATE TABLA
 33    SET PJ = PJ + 1,
 34        GF = GF + G2,
 35        GC = GC + G1,
 36        GD = GD + (G2 - G1),
 37        PG = PG + CASE WHEN G2 > G1 THEN 1 ELSE 0 END,
 38        PE = PE + CASE WHEN G2 = G1 THEN 1 ELSE 0 END,
 39        PP = PP + CASE WHEN G2 < G1 THEN 1 ELSE 0 END,
 40        PTOS = PTOS + CASE WHEN G2 > G1 THEN 3 WHEN G2 = G1 THEN 1 ELSE 0 END
 41    WHERE ID_EQU = V_SEL2;
 42
 43  END;
 44  /

Procedimiento creado.
```

VALIDACION DEL PROCEDURE (F) Finalizado

```
SQL> SELECT NUM_GOL_SEL1, NUM_GOL_SEL2 FROM PARTIDOS WHERE NUM_PAR = 1;

NUM_GOL_SEL1 NUM_GOL_SEL2
------------ ------------
           1            1

SQL> EXEC FINALIZAR_PARTIDO(1);

Procedimiento PL/SQL terminado correctamente.
```

```
SQL>    SELECT ESTADO_PAR FROM PARTIDOS WHERE NUM_PAR = 1;

E
-
F
```

2.8 Conclusiones

- El desarrollo del presente proyecto ha permitido confirmar la eficacia de un diseño de base de datos relacional para la gestión integral de un evento de la magnitud de un Mundial de Fútbol. Se logró estructurar de manera coherente la información esencial de selecciones, jugadores, árbitros y partidos, lo que facilitará un control detallado de los eventos ocurridos durante los encuentros. La implementación de triggers demostró ser fundamental para asegurar la integridad y consistencia de los datos. Estos mecanismos automatizados permitieron hacer cumplir reglas de negocio críticas, como la validación del número de titulares por equipo, la prevención de la doble participación de jugadores y árbitros en roles conflictivos, y la aplicación de sanciones disciplinarias, como la inhabilitación por expulsión. Asimismo, la creación de procedimientos almacenados para el control del estado de los partidos (inicio, anulación de goles y finalización) optimizó la gestión de los eventos en tiempo real. Particularmente, el procedimiento para la anulación de goles por situaciones de VAR y la actualización automática de la tabla de posiciones al finalizar un partido, resaltan la capacidad de automatización del sistema para reflejar con precisión los resultados del torneo. En síntesis, el sistema propuesto cumple con los objetivos planteados, proporcionando una solución robusta y eficiente para la planificación y ejecución de los partidos en un Mundial de Fútbol, garantizando la fiabilidad y la validez de la información registrada.

2.9 Referencias bibliográficas

- K. Owens, *Programming Oracle Triggers and Stored Procedures*. Prentice Hall Professional, 2004.
- F. Ress, "Oracle database packages, procedures, functions, and triggers: When and how to use them," Argonne National Lab.(ANL), Argonne, IL (United States), Tech. Rep. ANL/MCS/CP-86937, 1995.
- J. Juneau and M. Arena, "Triggers," in *Oracle and PL/SQL Recipes: A Problem-Solution Approach*, Springer, 2010, pp. 93–117.