



	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12

Entregar un enlace a github al producto desarrollado en cada una de las fases del proyecto, y un archivo zip con el proyecto git.

El Readme de Github debe incluir una descripción de la GENERACIÓN DE CÓDIGO DE COMPONENTES Y PROCEDIMIENTOS (CSI-2)

CÓDIGO DE COMPONENTES
<pre> START TRANSACTION; CREATE TABLE IF NOT EXISTS Gimnasio (Direccion VARCHAR(45) NOT NULL, Nombre VARCHAR(20) NOT NULL, PRIMARY KEY (Nombre)) ; CREATE TABLE IF NOT EXISTS Sala (Numero INT NOT NULL, Capacidad INT NOT NULL, Gimnasio_Nombre VARCHAR(20) NOT NULL, PRIMARY KEY (Numero, Gimnasio_Nombre), CONSTRAINT fk_Sala_Gimnasio FOREIGN KEY (Gimnasio_Nombre) REFERENCES Gimnasio (Nombre) ON DELETE NO ACTION ON UPDATE NO ACTION) ; CREATE TABLE IF NOT EXISTS Actividad (Nombre VARCHAR(30) NOT NULL, Precio FLOAT NOT NULL, Horario VARCHAR(20) NOT NULL, Sala_Numero INT NOT NULL, Sala_Gimnasio_Nombre VARCHAR(20) NOT NULL, PRIMARY KEY (Nombre, Sala_Numero, Sala_Gimnasio_Nombre), CONSTRAINT fk_Actividad_Sala1 FOREIGN KEY (Sala_Numero , Sala_Gimnasio_Nombre) REFERENCES Sala (Numero , Gimnasio_Nombre) ON DELETE NO ACTION ON UPDATE NO ACTION) </pre>

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12


```

;

CREATE TABLE IF NOT EXISTS Material (
    Nombre VARCHAR(30) NOT NULL,
    Numero INT NOT NULL,
    Cantidad INT NOT NULL,
    Sala_Numero INT NOT NULL,
    Sala_Gimnasio_Nombre VARCHAR(20) NOT NULL,
    Actividad_Nombre VARCHAR(30) NOT NULL,
    PRIMARY KEY (Numero, Sala_Numero, Sala_Gimnasio_Nombre, Actividad_Nombre),
    CONSTRAINT fk_Material_Sala1
        FOREIGN KEY (Sala_Numero , Sala_Gimnasio_Nombre)
        REFERENCES Sala (Numero , Gimnasio_Nombre)
        ON DELETE NO ACTION
        ON UPDATE CASCADE,
    CONSTRAINT fk_Material_Actividad1
        FOREIGN KEY (Actividad_Nombre)
        REFERENCES Actividad (Nombre)
        ON DELETE NO ACTION
        ON UPDATE CASCADE)
;

CREATE TABLE IF NOT EXISTS Averia (
    Numero INT NOT NULL,
    Descrpcion TEXT NULL,
    Material_Numero INT NOT NULL,
    Material_Sala_Numero INT NOT NULL,
    Material_Sala_Gimnasio_Nombre VARCHAR(20) NOT NULL,
    PRIMARY KEY (Numero, Material_Numero, Material_Sala_Numero,
Material_Sala_Gimnasio_Nombre),
    CONSTRAINT fk_Averia_Material1
        FOREIGN KEY (Material_Numero , Material_Sala_Numero ,
Material_Sala_Gimnasio_Nombre)
        REFERENCES Material (Numero , Sala_Numero , Sala_Gimnasio_Nombre)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION)
;

```


	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12

```
CREATE TABLE IF NOT EXISTS Persona (
  DNI CHAR(9) NOT NULL,
  Nombre VARCHAR(20) NULL,
  Apellidos VARCHAR(30) NULL,
  Fecha_Nacimiento DATE NULL,
  Direccion VARCHAR(40) NULL,
  PRIMARY KEY (DNI))
;
```

```
CREATE TABLE IF NOT EXISTS Cliente (
  FechaInscripcion DATE NOT NULL,
  FechaBaja DATE NOT NULL,
  Persona_DNI CHAR(9) NOT NULL,
  PRIMARY KEY (Persona_DNI),
  CONSTRAINT fk_Cliente_Personal
    FOREIGN KEY (Persona_DNI)
    REFERENCES Persona (DNI)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
;
```

```
CREATE TABLE IF NOT EXISTS Staff (
  Salario FLOAT NOT NULL,
  Persona_DNI CHAR(9) NOT NULL,
  PRIMARY KEY (Persona_DNI),
  CONSTRAINT fk_Staff_Personal
    FOREIGN KEY (Persona_DNI)
    REFERENCES Persona (DNI)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
;
```

```
CREATE TABLE IF NOT EXISTS Personal (
  Puesto VARCHAR(15) NOT NULL,
  Staff_Persona_DNI CHAR(9) NOT NULL,
  PRIMARY KEY (Staff_Persona_DNI),
  CONSTRAINT fk_Personal_Staff1
    FOREIGN KEY (Staff_Persona_DNI)
```

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12


```

REFERENCES Staff (Persona_DNI)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
;

CREATE TABLE IF NOT EXISTS Monitor (
    Especialidad VARCHAR(15) NOT NULL,
    Staff_Persona_DNI CHAR(9) NOT NULL,
    Actividad_Nombre VARCHAR(30) NOT NULL,
    Actividad_Sala_Numero INT NOT NULL,
    Actividad_Sala_Gimnasio_Nombre VARCHAR(20) NOT NULL,
    PRIMARY KEY (Staff_Persona_DNI, Actividad_Nombre, Actividad_Sala_Numero,
Actividad_Sala_Gimnasio_Nombre),
    CONSTRAINT fk_Monitor_Staff1
    FOREIGN KEY (Staff_Persona_DNI)
    REFERENCES Staff (Persona_DNI)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
    CONSTRAINT fk_Monitor_Actividad1
    FOREIGN KEY (Actividad_Nombre , Actividad_Sala_Numero ,
Actividad_Sala_Gimnasio_Nombre)
    REFERENCES Actividad (Nombre , Sala_Numero , Sala_Gimnasio_Nombre)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
;

CREATE TABLE IF NOT EXISTS Torneo (
    Nombre VARCHAR(20) NOT NULL,
    Ganador VARCHAR(20) NULL,
    Staff_Persona_DNI CHAR(9) NOT NULL,
    PRIMARY KEY (Nombre),
    CONSTRAINT fk_Torneo_Staff1
    FOREIGN KEY (Staff_Persona_DNI)
    REFERENCES Staff (Persona_DNI)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
;

```


	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12

```

CREATE TABLE IF NOT EXISTS Actividad_has_Cliente (
  Actividad_Nombre VARCHAR(30) NOT NULL,
  Actividad_Sala_Numero INT NOT NULL,
  Actividad_Sala_Gimnasio_Nombre VARCHAR(20) NOT NULL,
  Cliente_Persona_DNI CHAR(9) NOT NULL,
  PRIMARY KEY (Actividad_Nombre, Actividad_Sala_Numero,
Actividad_Sala_Gimnasio_Nombre, Cliente_Persona_DNI),
  CONSTRAINT fk_Actividad_has_Cliente_Actividad1
    FOREIGN KEY (Actividad_Nombre , Actividad_Sala_Numero ,
Actividad_Sala_Gimnasio_Nombre)
    REFERENCES Actividad (Nombre , Sala_Numero , Sala_Gimnasio_Nombre)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT fk_Actividad_has_Cliente_Cliente1
    FOREIGN KEY (Cliente_Persona_DNI)
    REFERENCES Cliente (Persona_DNI)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
;

CREATE TABLE IF NOT EXISTS Actividad_has_Monitor (
  Actividad_Nombre VARCHAR(30) NOT NULL,
  Actividad_Sala_Numero INT NOT NULL,
  Actividad_Sala_Gimnasio_Nombre VARCHAR(20) NOT NULL,
  Monitor_Staff_Persona_DNI CHAR(9) NOT NULL,
  PRIMARY KEY (Actividad_Nombre, Actividad_Sala_Numero,
Actividad_Sala_Gimnasio_Nombre, Monitor_Staff_Persona_DNI),
  CONSTRAINT fk_Actividad_has_Monitor_Actividad1
    FOREIGN KEY (Actividad_Nombre , Actividad_Sala_Numero ,
Actividad_Sala_Gimnasio_Nombre)
    REFERENCES Actividad (Nombre , Sala_Numero , Sala_Gimnasio_Nombre)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT fk_Actividad_has_Monitor_Monitor1
    FOREIGN KEY (Monitor_Staff_Persona_DNI)
    REFERENCES Monitor (Staff_Persona_DNI)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)

```

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12


```

;

CREATE TABLE IF NOT EXISTS Torneo_has_Persona (
  Torneo_Nombre VARCHAR(20) NOT NULL,
  Persona_DNI CHAR(9) NOT NULL,
  PRIMARY KEY (Torneo_Nombre, Persona_DNI),
  CONSTRAINT fk_Torneo_has_Persona_Torneo1
    FOREIGN KEY (Torneo_Nombre)
    REFERENCES Torneo (Nombre)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT fk_Torneo_has_Persona_Persona1
    FOREIGN KEY (Persona_DNI)
    REFERENCES Persona (DNI)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
;

COMMIT;

```

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12

CÓDIGO DE PROCEDIMIENTOS DE OPERACIÓN Y SEGURIDAD

TRIGGER 1

Trigger para comprobar que una sala no puede tener un número de personas mayor que su capacidad.

```
CREATE OR REPLACE FUNCTION comprobar_aforo_sala() RETURNS TRIGGER AS
$comprobar_aforo_sala$
BEGIN
    IF (SELECT COUNT(DISTINCT DNI) FROM SALA > Capacidad) THEN
        RAISE EXCEPTION 'No se puede añadir el cliente a la actividad porque la sala no tiene
más capacidad ';
    END IF;
    RETURN NEW;
END;
$comprobar_numero_generos$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trigger_comprobar_aforo_sala_before_insert BEFORE INSERT ON
ACTIVIDAD_HAS_CLIENTE
FOR EACH ROW EXECUTE PROCEDURE comprobar_aforo_sala();
```

```
CREATE TRIGGER trigger_comprobar_aforo_sala_before_insert BEFORE UPDATE ON
SERIE_PERTENECE_GENERO
FOR EACH ROW EXECUTE PROCEDURE comprobar_aforo_sala();
```


TRIGGER 2

Trigger para comprobar que una persona no puede ser cliente y staff a la vez

```
CREATE OR REPLACE FUNCTION comprobar_cliente_staff() RETURNS TRIGGER AS
$comprobar_cliente_staff$
BEGIN
    IF (SELECT DNI FROM CLIENTE IN (SELECT DNI FROM STAFF) THEN
        RAISE EXCEPTION No puede ser un cliente, ya que esa persona forma parte del staff ';
    END IF;
    RETURN NEW;
END;
$comprobar_cliente_staff$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trigger_comprobar_cliente_staff_before_insert BEFORE INSERT ON
CLIENTE
FOR EACH ROW EXECUTE PROCEDURE comprobar_cliente_staff();
```

```
CREATE TRIGGER trigger_comprobar_cliente_staff_before_insert BEFORE UPDATE ON
CLIENTE FOR EACH ROW EXECUTE PROCEDURE comprobar_cliente_staff();
```

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12

TRIGGER 3

Trigger para comprobar que una persona no puede ser cliente y staff a la vez

/* Create function 'comprobar_staff_cliente' */

```
CREATE OR REPLACE FUNCTION comprobar_staff_cliente() RETURNS TRIGGER AS
$comprobar_staff_cliente$
BEGIN
    IF (SELECT DNI FROM STAFF IN (SELECT DNI FROM CLIENTE) THEN
        RAISE EXCEPTION No puede ser un staff, ya que esa persona es un cliente ';
    END IF;
    RETURN NEW;
END;
$comprobar_staff_cliente $ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trigger_comprobar_staff_cliente_before_insert BEFORE INSERT ON
STAFF
FOR EACH ROW EXECUTE PROCEDURE comprobar_staff_cliente();
```

```
CREATE TRIGGER trigger_comprobar_staff_cliente_before_insert BEFORE UPDATE ON
STAFF FOR EACH ROW EXECUTE PROCEDURE comprobar_staff_cliente();
```


TRIGGER 4

Trigger para comprobar que un monitor no puede inscribirse en la actividad que imparte

```
CREATE OR REPLACE FUNCTION comprobar_monitor_actividad() RETURNS TRIGGER AS
$comprobar_monitor_actividad$
BEGIN
    IF (SELECT DNI FROM ACTIVIDAD_HAS_CLIENTE IN (SELECT DNI FROM
ACTIVIDAD_HAS_MONITOR) THEN
        RAISE EXCEPTION No puede participar en la actividad ya que la imparte ';
    END IF;
    RETURN NEW;
END;
$comprobar_monitor_actividad $ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trigger_comprobar_monitor_actividad_before_insert BEFORE INSERT ON
STAFF
FOR EACH ROW EXECUTE PROCEDURE comprobar_monitor_actividad();
```

```
CREATE TRIGGER trigger_comprobar_monitor_actividad_before_insert BEFORE UPDATE ON
STAFF FOR EACH ROW EXECUTE PROCEDURE comprobar_monitor_actividad();
```


	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)		BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:		Generación de código
	Autor:		
Versión: x		Tiempo invertido:	Fecha : 17/9/12

TRIGGER 5

Trigger para comprobar que un torneo tiene mínimo dos participantes para que se realice.

```

CREATE OR REPLACE FUNCTION comprobar_participantes_torneo() RETURNS TRIGGER
AS $comprobar_participantes_torneo$
BEGIN
    IF (SELECT COUNT(DISTINCT DNI) FROM TORNEO < 2) THEN
        RAISE EXCEPTION 'No se puede celebrar el torneo porque hay menos de dos personas ';
    END IF;
    RETURN NEW;
END;
$comprobar_participantes_torneo$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_comprobar_participantes_torneo_before_insert BEFORE INSERT
ON TORNEO_HAS_PERSONA
FOR EACH ROW EXECUTE PROCEDURE comprobar_aforo_sala();

CREATE TRIGGER trigger_comprobar_participantes_torneo_before_update BEFORE UPDATE
ON TORNEO_HAS_PERSONA
FOR EACH ROW EXECUTE PROCEDURE comprobar_aforo_sala();

```