EJERCICIO 1. (2½ Puntos) Modelado entidad relación

Realizar un modelo conceptual de datos mediante la técnica del modelo **Entidad Relación de Chen** teniendo en cuenta la siguiente descripción:

La **Fundación SCP** es una organización clandestina y global que opera al margen de cualquier jurisdicción gubernamental, encargada de gestionar objetos, entes y fenómenos anómalos que desafían las leyes naturales. Bajo su misión tripartita de «Asegurar, Contener, Proteger», la organización intercepta estas anomalías para evitar que caigan en manos ajenas, limita su influencia mediante estrictos protocolos de aislamiento y protege a la humanidad de sus efectos, priorizando el estudio científico y recurriendo a la neutralización solo cuando el peligro resulta incontrolable para preservar la normalidad de la vida cotidiana.

La Fundación mantiene una extensa base de datos que contiene información sobre anomalías. La base de datos principal de la Fundación comprende resúmenes de tales anomalías y procedimientos de emergencia que mantengan o restablezcan la contención de seguridad en caso de una brecha en la contención u otros eventos.

Las anomalías adoptan múltiples formas, sean objetos, entes, ubicaciones o fenómenos independientes. Estas anomalías son categorizadas dentro de una de las varias clases de objeto y pueden ser contenidas en una de las muchas instalaciones de la Fundación o contenidas sobre el terreno cuando su traslado se considera impracticable. Para garantizar su identificación, cada anomalía cuenta con un código y una descripción. Es indispensable conocer la fecha de contención en el caso de realizarse en una instalación de la Fundación. Tenga en cuenta que las anomalías pueden estar relacionadas entre sí, por lo que será necesario llevar un registro de qué anomalías referencian a otras. Además, para llevar una correcta trazabilidad, hay que conocer en todo momento, en qué contenedor (código alfanumérico) de la instalación se ha asegurado la anomalía.

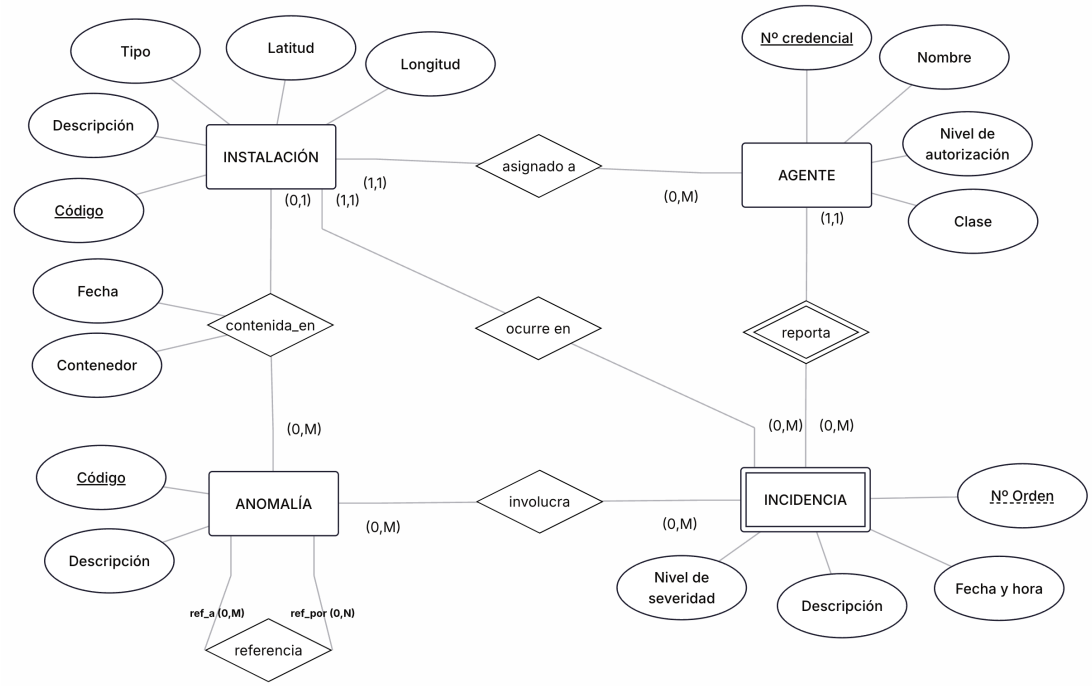
Las instalaciones de la Fundación están distribuidas a lo largo del planeta en posiciones definidas por latitud y longitud. La identificación de las instalaciones recae en un código alfanumérico. Importante destacar el tipo de instalación (sitio, área, puesto de avanzada o puesto de observación) así como su descripción.

El capital humano es el recurso más valioso y, a la vez, el más prescindible de la organización. La base de datos debe registrar a todo el Personal vinculado, ya sea personal interno de la Fundación o consultores externos de agencias gubernamentales (como los agentes *Fox Mulder* y *Dana Scully*, recientemente adscritos bajo la directiva de cooperación clasificada). De cada miembro del personal se requiere almacenar un número de credencial único, su nombre completo, su nivel de Autorización de Seguridad (un número del 0 al 5) y su clase (A, B, C, D o E). Es fundamental saber a qué instalación está asignado cada miembro del personal como su base operativa principal; sin embargo, tenga en cuenta que, por motivos de seguridad, un empleado debe estar asignado a una única instalación en un momento dado.

Finalmente, la naturaleza impredecible de las anomalías genera Incidencias. Estas situaciones van desde brechas de contención hasta fenómenos inexplicables investigados por el FBI y transferidos a la Fundación (los denominados «Expedientes X»). El sistema debe ser capaz de relacionar qué Personal reportó la incidencia (un único responsable por informe) y qué anomalías estuvieron involucradas en la misma (puede haber varias anomalías causantes de un mismo incidente, o ninguna si es un evento aún no clasificado). Cada incidencia incluye un número de orden incremental en función del Agente que la reporta, y debe incluir la fecha y hora del suceso, una descripción detallada y un nivel de severidad. Asimismo, es necesario conocer en qué instalación ocurrió la incidencia para facilitar los protocolos de limpieza.

Se pide realizar un **modelo entidad-relación de Chen** justificando las **cardinalidades mínimas** de al menos dos relaciones (**ambos lados de la relación**).

Solución propuesta:



Cardinalidades mínimas:

- La cardinalidad mínima de incidencia con respecto a agente es 1, ya que se trata de una entidad débil y, por tanto, hay que conocer en todo momento qué agente la ha reportado. Por otra parte, la cardinalidad mínima de agente con respecto a incidencia es 0, pues es posible que un determinado agente no reporte incidencia alguna.
- La cardinalidad mínima de instalación con respecto a agente es 1, puesto que por seguridad un agente debe estar asignado a una única instalación. Por

otra parte, la cardinalidad mínima de agente con respecto a instalación es 0, puesto que puede existir una instalación que no tenga agentes asignados.

EJERCICIO 2. (3 Puntos) Dado el siguiente modelo relacional[1]:

`CONTINENTE (idC, nombreC)`

`IDIOMA (idI, nombreI, abreviatura)`

`PAIS (idP, nombreP, abreviatura, capital, moneda, poblacion, extension, costa,` *idC*$^{\text{FK}}$`)`

`SE_HABLA (`*idP*$^{\text{FK}}$`,` *idI*$^{\text{FK}}$`, num_hablantes)`

`TIENE_TURISMO (`*idP_origen*$^{\text{FK}}$`,` *idP_destino*$^{\text{FK}}$`, año, num_turistas)`

Se pide dar respuesta a las siguientes cuestiones:

(a) (½ Punto) Resolver mediante álgebra relacional:"*Obtener la abreviatura de los idiomas que solamente se hablan en países con población menor a 100.000 habitantes*".

Solución propuesta:

$\Pi_{abreviatura}[IDIOMA \bowtie ($
$\Pi_{idI}[\sigma_{\text{poblacion}<100.000}(SE\_HABLA \bowtie PAIS)]$
$-$
$\Pi_{idI}[\sigma_{\text{poblacion}>=100.000}(SE\_HABLA \bowtie PAIS)]$
$)]$

(b) (½ Punto) Resolver mediante álgebra relacional: "*Obtener el id y nombre de aquellos países del continente Europeo donde se hablan todos los idiomas existentes*".

Solución propuesta:
$\Pi_{idP,nombreP,idI}(\sigma_{\text{nombreC}>=\text{'Europa'}}(CONTINENTE \bowtie PAIS$
$\bowtie SE\_HABLA)) \div \Pi_{idI}(IDIOMA)$

(c) (½ Punto) Resolver mediante SQL la siguiente consulta: "*Obtener el nombre del país y la extensión de aquel país que tiene la mayor extensión de tierra dentro del continente Asiático*".

Solución propuesta:

```
SELECT nombreP, extension
FROM PAIS INNER JOIN CONTINENTE ON PAIS.idC = CONTINENTE.idC
WHERE nombreC = 'Asia'
AND extension >= ALL (SELECT extension
                      FROM PAIS INNER JOIN CONTINENTE ON PAIS
                          .idC = CONTINENTE.idC
                      WHERE nombreC = 'Asia');
```

[1] Tenga en cuenta que las claves primarias aparecen subrayadas mientras que las claves foráneas aparece en *cursiva* junto con el superíndice FK y comparten nombre con la clave primaria a la que referencian.

(d) (½ Punto) Resolver mediante SQL la siguiente consulta: "*Obtener el nombre de los países que, en el año 2023, fueron tanto un destino que recibió más de 9000 turistas, como un origen que del que salieron más de 9000 turistas*".

Solución propuesta:

```
SELECT nombreP
FROM PAIS
WHERE idP IN (SELECT idP_destino FROM TIENE_TURISMO
              WHERE año = 2023
              GROUP BY idP_destino
              HAVING SUM(num_turistas) > 9000)
AND idP IN (SELECT idP_origen FROM TIENE_TURISMO
            WHERE año = 2023
            GROUP BY idP_origen
            HAVING SUM(num_turistas) > 9000);
```

(e) (1 Punto) Resolver mediante SQL la siguiente consulta: "*Obtener, para aquellos países en los que únicamente se habla un idioma y que reciben turistas de otros países, su nombre, el nombre del continente al que pertenece, así como la cantidad total de países diferentes de los que reciben turistas*".

Solución propuesta:

```
SELECT PAIS.idP, PAIS.nombreP, CONTINENTE.nombreC,
       COUNT(DISTINCT TIENE_TURISMO.idP_origen) AS
          NumPaisesOrigen
FROM PAIS INNER JOIN CONTINENTE ON PAIS.idC = CONTINENTE.idC
          INNER JOIN TIENE_TURISMO ON PAIS.idP =
             TIENE_TURISMO.idP_destino
WHERE PAIS.idP IN (SELECT idP
                   FROM SE_HABLA
                   GROUP BY idP
                   HAVING COUNT(idI) = 1)
GROUP BY PAIS.idP, PAIS.nombreP, CONTINENTE.nombreC;
```

EJERCICIO 3. (2 ½ Puntos) Teniendo en cuenta el modelo relacional del ejercicio anterior, complete las siguientes preguntas:

(a) (1 Punto) Se ha detectado una posible problemática dentro de la base de datos: se quiere evitar, mediante un **TRIGGER**, que el número de hablantes de un idioma en un país supere la población total de dicho país. Si esto ocurre, el TRIGGER debe impedir la inserción de nuevos valores y lanzar un error de aplicación con el mensaje "*El número de hablantes excede la población del país*". Codifica dicho TRIGGER.

Solución propuesta:

```
DELIMITER $$
CREATE OR REPLACE TRIGGER VALIDAR_HABLANTES
BEFORE INSERT ON SE_HABLA
FOR EACH ROW
BEGIN
    DECLARE v_poblacion INT DEFAULT 0;

    SELECT poblacion INTO v_poblacion
    FROM PAIS
    WHERE idP = NEW.idP;

    IF NEW.num_hablantes > v_poblacion THEN
        SIGNAL SQLSTATE "02000"
        SET MESSAGE_TEXT = "El numero de hablantes excede la
           poblacion del pais";
    END IF;
END$$
DELIMITER ;
```

(b) (1½ Puntos) En sus años de vida, José Antonio Labordeta siempre soñó con la capacidad de analizar los destinos a los que viajaba con su entrañable mochila. En honor a él, se quiere crear una tabla adicional:

`BITACORA_LABORDETA`(idLabordeta INT AUTO_INCREMENT, *idP*$^{\text{FK}}$ VARCHAR(100), *veredicto* VARCHAR(200))

Se pide: (1) crear la **tabla** `BITACORA_LABORDETA`, respetando la arquitectura mencionada anteriormente, (2) crear un **PROCEDIMIENTO** que lleve por nombre `ANALIZAR_DESTINOS_MOCHILA` que reciba como parámetro de entrada el identificador de un continente y analice todos los paises de dicho continente para rellenar la tabla `BITACORA_LABORDETA`. Para ello, se debe determinar el número de turistas que recibe cada pais y por cada país realice la inserción de datos necesaria para que el veredicto cumpla las siguientes condiciones:

- Si tiene más de 5.000.000 de turistas: "Demasiado ruido, ¡A la mierda!"
- Si tiene menos de 5.000.000 de turistas: "Buen sitio para sacar la navaja y el queso".
- Si no tiene turistas: "Aquí me quedo a contemplar la calma".

Solución propuesta:

```
-- PARTE 1
CREATE TABLE BITACORA_LABORDETA (
    idLabordeta int auto_increment,
    idP varchar(100),
    veredicto varchar(200),
    PRIMARY KEY (idLabordeta),
    CONSTRAINT FK_LABORDETA
        FOREIGN KEY (idP) REFERENCES Pais(idP)
);


-- PARTE 2
DELIMITER $$
CREATE PROCEDURE ANALIZAR_DESTINOS_MOCHILA (IN p_idC VARCHAR
   (100))
BEGIN

    DECLARE done INT DEFAULT FALSE;
    DECLARE v_total_turistas INT;
    DECLARE v_veredicto VARCHAR(200);

    DECLARE v_idP_pais INT; --Depende del tipo de idP

    DECLARE c_paises CURSOR FOR SELECT p.idP
                                FROM PAIS p
                                    JOIN CONTINENTE c ON p.
                                       idC = c.idC
                                WHERE c.idC = p_idC;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

    OPEN c_paises;
    read_loop: LOOP
        FETCH c_paises INTO v_idP_pais;

        IF done THEN
            LEAVE read_loop;
        END IF;

        SELECT SUM(num_turistas) INTO v_total_turistas
        FROM TIENE_TURISMO
        WHERE idP_destino = v_idP_pais;


        IF v_total_turistas IS NULL OR v_total_turistas = 0
           THEN
            SET v_veredicto = 'Aqui me quedo a contemplar la
               calma';
            SET done = false;
        ELSEIF v_total_turistas > 5000000 THEN
            SET v_veredicto = 'Demasiado ruido, A la mierda!'
               ;
        ELSE
```

```
            SET v_veredicto = 'Buen sitio para sacar la
                navaja y el queso';
        END IF;

        INSERT INTO BITACORA_LABORDETA (idP, veredicto)
        VALUES (v_idP_pais, v_veredicto);

    END LOOP;
    CLOSE c_paises;
END$$
DELIMITER ;
```

EJERCICIO 4. (1 Punto) Tras décadas de negociaciones y visitas diplomáticas, los habitantes de *Utopía del Norte* y *Utopía del Sur* han logrado la unificación del país en *Utopía*. Como consecuencia de esta unificación, es necesario actualizar la base de datos del ejercicio 2 para: (1) renombrar *Utopía del Norte* (`idP = 23`) por *Utopía*, (2) agregar los datos de número de hablantes y número de turistas de *Utopía del Sur* (*idP = 42*) a los de *Utopía del Norte* (ahora llamado *Utopía)*, y (3) eliminar *Utopía del Sur*. Ambos países acordaron de mutuo acuerdo que la unificación se llevara a cabo con carácter retroactivo a fecha 1 de enero de 2026, por lo que la agregación de los datos referentes al turismo se debe realizar desde dicha fecha en adelante. Se pide codificar una transacción lleve a cabo las actualizaciones anteriormente mencionadas.

Solución propuesta:

```
START TRANSACTION;

UPDATE pais
SET nombreP = 'Utopia',
    poblacion = poblacion + (SELECT poblacion
                             FROM pais
                             WHERE idP = 42)
WHERE idP = 23;

UPDATE tiene_turismo
SET num_turistas = num_turistas + (SELECT num_turistas
                                   FROM tiene_turismo
                                   WHERE ipP_origen = 42
                                     AND año = 2026)
WHERE idP_origen = 23
  AND año = 2026;

UPDATE tiene_turismo
SET num_turistas = num_turistas + (SELECT num_turistas
                                   FROM tiene_turismo
                                   WHERE ipP_destino = 42
                                     AND año = 2026)
WHERE ipP_destino = 23
  AND año = 2026;

UPDATE se_habla sh1
INNER JOIN se_habla sh2 ON sh1.idI = sh2.idI
SET sh1.num_hablantes = sh1.num_hablantes + sh2.num_hablantes
WHERE sh1.idP = 23 AND sh2.idP = 42;

INSERT INTO se_habla (idP, idI, num_hablantes)
SELECT 23, idI, num_hablantes
FROM se_habla
WHERE idP = 42
  AND idI NOT IN (SELECT idI
                  FROM se_habla
                  WHERE idP = 23);
```

```
DELETE FROM tiene_turismo
WHERE ipP_destino = 42
   OR ipP_origen = 42;

DELETE FROM se_habla
WHERE idP = 42;

DELETE FROM pais
WHERE idP = 42;

COMMIT;
```

EJERCICIO 5. (1 Punto) Se dispone de un programa Java para la gestión de la base de datos del ejercicio 2 que, en su fichero `pom.xml`, ha cargado la siguiente dependencia:

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.18</version>
</dependency>
```

Este programa dispone de una clase `Main` que comienza del siguiente modo:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class Main {

    Connection conn;

    public static void main(String[] args) throws Exception {
        Class.forName("com.mysql.cj.jdbc.Driver").newInstance();
        conn = DriverManager.getConnection("jdbc:mysql://localhost/world?" +
            "user=god&password=str0ngP4ssw0rd!");

        insetarIdioma("Klingon", "KLI");

        int num = getNumTotalHablantes(1, 2);
        System.out.println("Número de hablantes del continente 1: " + num);
    }
```

Completa la codificación de los métodos `insetarIdioma` y `getNumTotalHablantes` de acuerdo a los comentarios de su JavaDoc.

(a) (½ Punto) Completa el siguiente método:

```
/**
 * Permite insertar un idioma en la base de datos. El identificador del idioma
 * (idI) se genera automáticamente.
 * @param nombreI Nombre del idioma
 * @param abreviatura Abreviatura del idioma
 * @throws Arroja cualquier excepción que se pueda producir
 */
public static void insetarIdioma (String nombreI, String abreviatura)
    throws Exception {
```

Solución propuesta:

```
PreparedStatement stmt = conn.prepareStatement(
    "INSERT INTO idioma (nombreI, abreviatura) VALUES (?, ?)");
stmt.setString(1, nombreI);
stmt.setString(2, abreviatura);
stmt.executeUpdate();
stmt.close();
```

```
}
```

(b) (½ Punto) Completa el siguiente método:

```
/**
 * Obtiene el número total de hablantes de un idioma en un continente.
 * @idC Identificador del continente.
 * @idI Identificador del idioma.
 * @return Entero con el número total de hablantes.
 * @throws Arroja cualquier excepción que se pueda producir
 */
public static int getNumTotalHablantes (int idC, int idI) throws Exception {
```

Solución propuesta:

```
PreparedStatement stmt = conn.prepare("SELECT SUM(num_hablantes)" +
"FROM se_habla INNER JOIN pais ON pais.idP = se_habla.idP" +
"WHERE idC = ? AND idI = ?");

stmt.setInt(1, idC);
stmt.setInt(2, idI);

ResultSet rs = stmt.executeQuery();
rs.next();
int numHablantes = rs.getInt(1);

rs.close();
stmt.close();

return numHablantes;
```

```
}
```

Nota Importante

Este ejercicio contiene cuestiones relativas a la práctica de la asignatura. Aunque este ejercicio no computa en la nota del examen, representa el 20 % de la nota de prácticas. Será requisito indispensable obtener una calificación mínima de 3 sobre 10 en el mismo para que la nota de prácticas se agregue a la calificación final de la convocatoria ordinaria. Todos los subapartados de este ejercicio tienen la misma puntuación (i.e. 1/3 de la nota).

EJERCICIO 6. Para las siguientes afirmaciones relativas a la **práctica de la asignatura**, indica si son verdaderas o falsas, justificando tu respuesta con argumentos claros y precisos. La falta de justificación invalida la respuesta y no será considerada para la puntuación:

(a) La relación entre doctores y departamentos se modela únicamente con la tabla `affiliated_with` (con `physicianid`, `departmentid` y `primary_affiliation`), sin necesidad de ningún atributo adicional en la tabla `deparment`.

Solución propuesta:
Falso. Aunque la tabla `affiliated_with` modela correctamente la adscripción múltiple y el departamento principal (con `primary_affiliation`), también existe el atributo `head` (clave foránea) en la tabla `deparment` que representa específicamente al director de cada departamento. Un doctor puede dirigir varios departamentos, pero cada departamento tiene un único director.

(b) La tabla `trained_in` (con clave primaria compuesta `physicianid`, `treatmentid` y los atributos `certificationdate`, `certificationexpires`) representa una relación muchos-a-uno (N:1) entre doctores y procedimientos médicos.

Solución propuesta:
Falso. La clave primaria compuesta (`physicianid` + `treatmentid`) indica una relación muchos-a-muchos (N:M) entre doctores y procedimientos médicos, donde un doctor puede certificarse en múltiples procedimientos y un procedimiento puede ser realizado por múltiples doctores. Los atributos adicionales son propios de la relación misma, confirmando que requiere tabla intermedia para N:M.

(c) Los dos programas del apartado 3 de la práctica exportan datos de la vista a ficheros con formato `csv` o `xml`. Ambos requieren que en sus respectivos fichero `pom.xml` incluya la dependencia `mysql-connector-java`.

Solución propuesta:
Verdadero. Requiere explícitamente la dependencia `mysql-connector-java` para la conexión JDBC a MySQL.