Universidad Politécnica de Madrid · ETSI SISTEMAS INFORMÁTICOS	Apellidos:	
	Nombre:	
	DNI:	Num. mat.:

Normativa de examen

- No está permitido el uso de dispositivos móviles ni otros dispositivos electrónicos.
- Cada estudiante podrá disponer de un folio tamaño A4 con las anotaciones que considere oportunas para desarrollar su examen. El folio podrá contener texto por ambas caras.
- Durante el examen, el profesor podrán solicitar acreditar la identidad de los participantes en el mismo. Deberá tener en todo momento su Documento Nacional de Identidad y/o Carné de la UPM visible sobre la mesa.
- El examen deberá estar escrito en bolígrafo de color azul o negro.
- No se permite abandonar el aula de examen durante los primeros 15 minutos. Transcurrido este tiempo, no se permitirá entrar al examen.
- El examen tiene una duración máxima de **2.5 horas**.
- Justifique sus respuestas lo mejor posible indicando, si fuese necesario, los pasos realizados.
- Las calificaciones provisionales serán publicadas en el Moodle de la asignatura el viernes 14 de junio de 2024.
- La fecha para la revisión del examen se anunciará en el Moodle de la asignatura junto a la publicación de las calificaciones.

Bloque:	1	2	3	4	Total:
Puntos:	3	5½	1	½	10
Nota:					

1. **Bloque "Modelo Entidad/Relación y paso a tablas"**

   (a) (3 Puntos) La organización *Pokemon Masters Association* ha creado un sistema de registro para gestionar entrenadores y sus capturas en diversas localizaciones.

   Estas localizaciones tienen un nombre único, y pertenecen a una y solo una de las diez regiones, también cada una con su nombre único. **Todas** las regiones poseen localizaciones. Los entrenadores son identificados por su alias único y se guarda su fecha de nacimiento. Los pokemons tienen un número único, nombre, y tipo, que puede ser Acero, Agua, Bicho, Dragón, Eléctrico, Fantasma, Fuego, Hada, Hielo, Normal, Planta, Psíquico, Roca, Siniestro, Tierra, Veneno y Volador; por cierto, no se conoce por el momento ninguno tipo Hada, pero no se descarta encontrarlo en un futuro, así que se mantiene el tipo en la base de datos.

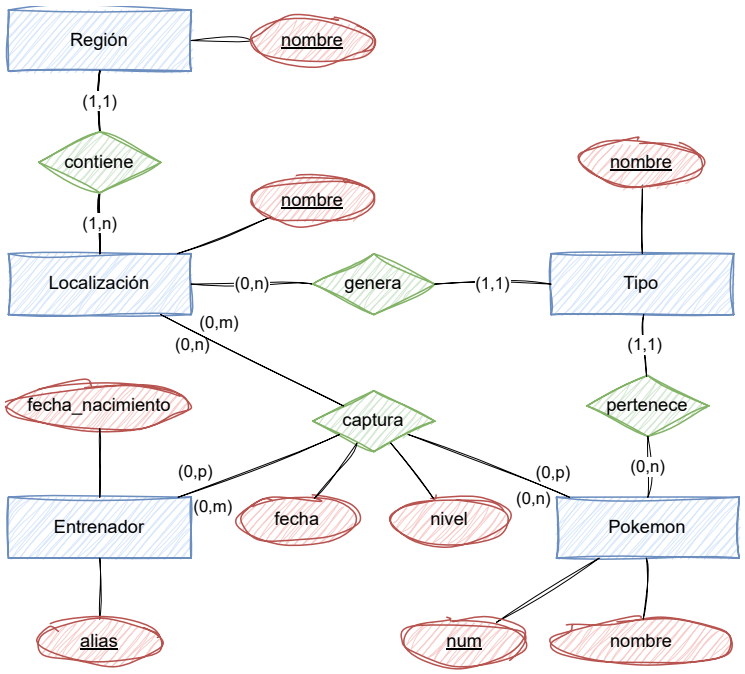
   Destacar que las localizaciones se caracterizan por atraer siempre a un tipo con mayor frecuencia, así que es interesante saber también cuál es ese tipo para cada localización; como antes, no se conoce ninguna localización que genere concretamente el tipo Hada.

   Cada vez que un entrenador captura un pokémon de un número determinado, esta se registra junto con la fecha de la captura y el nivel de dicho pokemon. Esta información será única para cada combinación de entrenador, pokemon, localización y región.

   Realizar:

   - Un modelo conceptual de datos mediante la técnica del **modelo Entidad-Relación de Chen** que modele el sistema de registro mencionado.

Solución propuesta:



2. **Bloque "Lenguaje SQL"**

   Dado la siguiente creación de tablas en SQL:

```sql
CREATE TABLE entrenadores (
    alias VARCHAR(50) UNIQUE NOT NULL,
    edad INT NOT NULL,
    PRIMARY KEY (alias)
);

CREATE TABLE pokemons (
    num INT UNIQUE NOT NULL,
    nombre VARCHAR(50) UNIQUE NOT NULL,
    tipo VARCHAR(50) NOT NULL,
    PRIMARY KEY (num)
);

CREATE TABLE localizaciones (
    nombre VARCHAR(50) UNIQUE NOT NULL,
    tipo   VARCHAR(50) NOT NULL,
    PRIMARY KEY (nombre)
);

CREATE TABLE capturas (
    entrenador VARCHAR(50) NOT NULL,
    pokemon INT UNIQUE NOT NULL,
    localizacion VARCHAR(50) NOT NULL,
    PRIMARY KEY (entrenador, pokemon, localizacion),
    CONSTRAINT fk_entrenador
        FOREIGN KEY (entrenador) REFERENCES entrenadores (alias),
    CONSTRAINT fk_pokemon
        FOREIGN KEY (pokemon) REFERENCES pokemons (num),
    CONSTRAINT fk_localizacion
        FOREIGN KEY (localizacion) REFERENCES localizaciones (nombre)
);
```

   Se pide:

   (a) (½ Punto) Escriba una consulta en SQL que devuelva el `alias` de los entrenadores que nunca han capturado un pokemon en la `localizacion` '*Monte Moon*'.

   Solución propuesta:

```sql
SELECT alias
FROM entrenadores
WHERE alias NOT IN (SELECT entrenador
                    FROM capturas
                    WHERE localizacion = 'Monte Moon')
```

   (b) (½ Punto) Escriba una consulta en SQL que devuelva el `nombre` de los pokemons que han sido capturados por los entrenadores '*Ash Ketchum*' y '*Gary Oak*'.

Solución propuesta:

```
SELECT nombre
FROM pokemons
WHERE num IN (SELECT pokemon
              FROM capturas
              WHERE entrenador = 'Ash Ketchum')
  AND num IN (SELECT pokemon
              FROM capturas
              WHERE entrenador = 'gary Oak')
```

(c) (½ Punto) Escriba una consulta en SQL que devuelva el `alias` de los entrenadores que se hayan capturado todos los pokemon de `tipo` '*fuego*'.

Solución propuesta:

```
SELECT alias
FROM entrenadores
WHERE NOT EXISTS (SELECT *
                  FROM pokemons
                  WHERE tipo = 'fuego'
                    AND NOT EXISTS (SELECT *
                                    FROM capturas
                                    WHERE capturas.pokemon =
                                            pokemons.num
                                      AND capturas.entrenador =
                                            entrenadores.nombre))
```

(d) (½ Punto) Escriba una consulta en SQL que devuelva el `nombre` de la localización o localizaciones donde más pokemons se han capturado.

Solución propuesta:

```
SELECT localizacion
FROM capturas
GROUP BY localizacion
HAVING COUNT(*) >= ALL (SELECT COUNT(*)
                        FROM capturas
                        GROUP BY localizacion);
```

(e) (½ Punto) Escriba una consulta en SQL que devuelva el `tipo` (o `tipos`) de pokemons del que existen menos pokemons.

Solución propuesta:

```sql
SELECT tipo
FROM pokemons
GROUP BY tipo
HAVING COUNT(*) <= ALL (SELECT COUNT(*)
                        FROM pokemons
                        GROUP BY tipo);
```

(f) (½ Punto) Escriba una consulta en SQL que devuelva el `alias` de los entrenadores que hayan capturado el pokemon denominado *New* en la localización '*Cueva Celeste*'.

Solución propuesta:

```sql
SELECT DISTINCT alias
FROM capturas
    INNER JOIN pokemons ON pokemons.num = captura.pokemon
WHERE nombre = 'New'
  AND localizacion = 'Cueva Celeste';
```

(g) (1 Punto) Escriba un procedimiento almacenado en SQL que dado el `nombre` de un pokemon, que se pasará como parámetro de entrada, liste por pantalla el número de veces que ha sido capturado en cada localización. Si un pokemon nunca ha sido capturado no es necesario que aparezca en el listado. El listado debe aparecer ordenador de mayor a menor número de capturas.

Solución propuesta:

```sql
DELIMITER $$
CREATE PROCEDURE num_veces_capturado (IN nombre_pokemon VARCHAR(50))
BEGIN
    SELECT localizacion, COUNT(*) AS num_capturas
    FROM capturas
        INNER JOIN pokemons ON pokemons.num = capturas.pokemon
    WHERE nombre = nombre_pokemon
    GROUP BY localizacion
    ORDER BY num_capturas DESC;
END$$
DELIMITER ;
```

(h) (1 Punto) Escriba un *trigger* en SQL que impida que los entrenadores menores de 16 años que añadan nuevas capturas de pokemons de `tipo` '*fuego*' para evitar sufrir quemaduras.

Solución propuesta:

```
DELIMITER $$
CREATE TRIGGER evita_fuego_16 BEFORE INSERT ON pokemons
FOR EACH ROW
BEGIN
    DECLARE e INTEGER;
    DECLARE t VARCHAR(50);

    SELECT edad INTO e
    FROM entrenadores
    WHERE alias = NEW.entrenador;

    SELECT tipo INTO t
    FROM pokemons
    WHERE num = NEW.pokemon;

    IF e < 16 AND tipo = 'fuego' THEN
        SIGNAL SQLSTATE '162342'
        SET MESSAGE_TEXT = 'Los entrenadores menores de 16 años
        no pueden capturar pokemons de tipo fuego';
    END IF;
END$$
DELIMITER ;
```

(i) (½ Punto) Cree un usuario denominado '*cartógrafo*' y otórguele permisos de consulta e inserción sobre la tabla `localizaciones`. A continuación, inserte una nueva localización denominada '*Pueblo Paleta*' de tipo '*ciudad*'.

Solución propuesta:

```
CREATE USER 'cartógrafo' IDENTIFIED BY '1234';

GRANT SELECT, INSERT ON localizaciones TO 'cartógrafo';

INSERT INTO localizaciones VALUES ("Pueblo Paleta", "Ciudad");
```

3. **Bloque "Acceso programático"**

   (a) (1 Punto) Tomando como punto de partida un `SCHEMA` de bases de datos basado en el modelo relacional del bloque anterior, complete los huecos del siguiente código fuente en `python` de tal manera que se satisfaga la funcionalidad indicada como comentarios.

```python
import mysql.connector

# establecimiento de conexion
mydb = mysql.connector.connect(
  host='localhost',
  user='root',
  password='root',
  database='pokemon_db',
)

cursor = mydb.cursor()

# Añadir entrenador
alias = 'Ash Ketchum'
edad = 10

cursor.execute(
    'INSERT INTO entrenadores _____________________________',
    (alias, edad)
)

mydb.commit()

# Listar los nombres y tipos de los pokémons capturados en una
# localización específica
localizacion_buscada = 'Bosque Verde'
consulta = '''
SELECT p.nombre, p.tipo
FROM pokemons p
JOIN capturas c ON p.num = c.pokemon
WHERE c.localizacion = %s
'''

cursor._____________(_________________________________________________)

for (nombre_pokemon, tipo_pokemon) in cursor:
    print(f'{nombre_pokemon} ({tipo_pokemon})')

# Cierre de recursos
cursor.close()
cnx.close()
mydb.close()
```

Solución propuesta:

```python
import mysql.connector

# establecimiento de conexion
mydb = mysql.connector.connect(
  host='localhost',
  user='root',
  password='root',
  database='pokemon_db',
)

cursor = mydb.cursor()

# Añadir entrenador
alias = 'Ash Ketchum'
edad = 10

cursor.execute(
    'INSERT INTO entrenadores (alias, edad) VALUES (%s, %s)',
    (alias, edad)
)

mydb.commit()

# Listar los nombres y tipos de los pokémons capturados en una localización específ
localizacion_buscada = 'Bosque Verde'
consulta = '''
SELECT p.nombre, p.tipo
FROM pokemons p
JOIN capturas c ON p.num = c.pokemon
WHERE c.localizacion = %s
'''

cursor.execute(consulta, (localizacion_buscada,))

for (nombre_pokemon, tipo_pokemon) in cursor:
    print(f'{nombre_pokemon} ({tipo_pokemon})')

# Cierre de recursos
cursor.close()
cnx.close()
mydb.close()
```

4. **Bloque "Modelo relacional"**

   (a) (½ Punto) ¿Qué pasos deberían darse para que la base de datos siguiente cumpliera la primera forma normal (1FN)?

ID	Nombre	Edad	Ciudad	Pokemon
1	Ash	15	Pueblo Paleta	Pikachu, Bulbasaur, Charmander
2	Misty	13	Ciudad Celeste	Psyduck, Starmie
3	Brock	16	Ciudad Plateada	Onix, Geodude, Zubat

Solución propuesta:
Actualmente existe más de un valor para el atributo `Pokemon`, por lo que se debería crear una nueva tabla que guardase `ID`, `Pokemon` para evitar que esto sucediera. En la tabla actual habría que eliminar la columna `Pokemon`.