	Bases de Datos. Curso 2019 / 2020		09 / 01 / 2020
	Apellidos: _____		Nombre: _____
	DNI: _____	Núm. matrícula: _____	Grupo: _____

Normativa de Examen

1. No está permitido el uso de dispositivos móviles ni otros dispositivos electrónicos.
2. No está permitido el uso de libros ni apuntes.
3. Durante el examen, los profesores 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.
4. Deberá escribir su nombre, con bolígrafo, en todas las hojas de las que consta el examen.
5. No se permite abandonar el aula de examen durante los primeros 15 minutos. Transcurrido este tiempo, no se permitirá entrar al examen.
6. La calificación máxima de este examen es 10 puntos.
7. El examen tiene una duración máxima de 2 horas 30 minutos.
8. Justifique sus respuestas lo mejor posible indicando, si es necesario, los pasos realizados.
9. Las calificaciones provisionales serán publicadas en el Moodle de la asignatura el día 16 de enero de 2020.
10. La revisión del examen se llevará a cabo el día 20 de enero de 2020.

Ejercicio 1 (2,0 puntos). El Ministerio de Medio Ambiente desea crear un Sistema de Información Geográfica (SIG) de acceso público a través de Internet. El sistema ofrecerá la siguiente información:

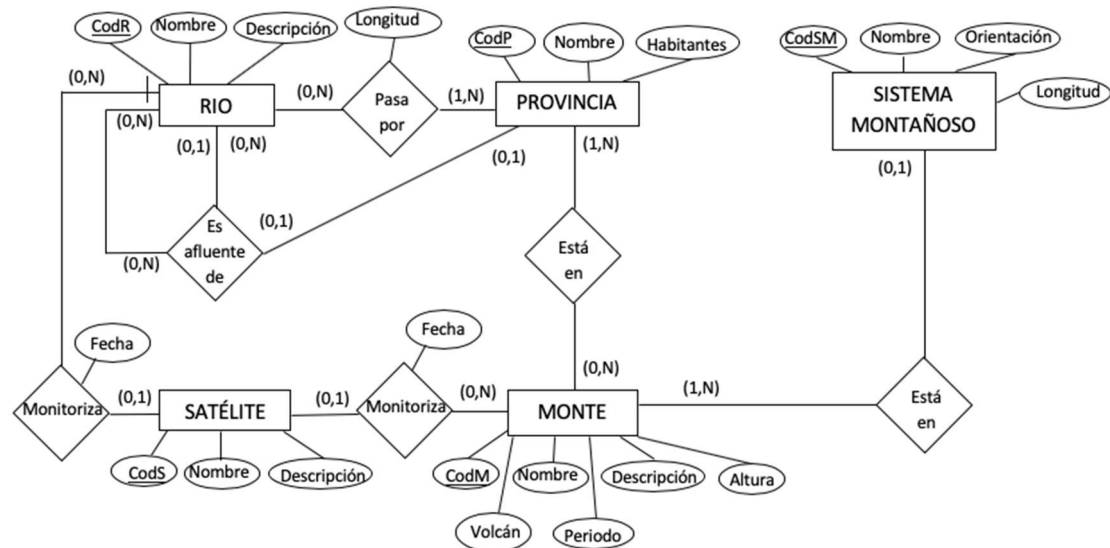
Datos referentes a ríos, afluentes, sistemas montañosos, montes y provincias donde se localizan. De los ríos se almacenará un código de río, nombre, descripción y longitud total. Para cada río, además, se almacenarán las provincias por las que pasa y la longitud del tramo de cada río para cada provincia bañada. De las provincias se almacenará un código de provincia, nombre y número de habitantes. Los ríos pueden a su vez ser afluentes de otros ríos. En este caso, se desea conocer de cual río son afluentes y la provincia en la que se unen.


En cuanto a los sistemas montañosos, se almacenará un código identificativo, el nombre, la orientación (norte, nordeste, etc.) y la longitud, así como la altura máxima y las provincias que ocupa. Los sistemas están formados por montes de los que se almacena un código, un nombre, descripción y altura. Hay que tener en cuenta que un monte sólo pertenecerá a un sistema. De los montes también se quiere almacenar la provincia o provincias en las que se encuentra, ya que hay casos en los que un monte es compartido por varias provincias. Los montes además pueden tener un origen volcánico o de plegamiento. En el caso de que su origen sea volcánico, se desea almacenar el tipo de volcán, y en caso de ser plegamiento, se almacenará el periodo geológico de dicho plegamiento.

Algunos ríos y montes son elementos geológicos monitorizados por satélite. De dichos elementos se desea almacenar la fecha en la que comienza su monitorización y el satélite que realiza el seguimiento. Un elemento monitorizado sólo puede serlo por un satélite y un satélite podrá monitorizar varios elementos. De los satélites se desea almacenar su número identificativo, nombre y descripción.

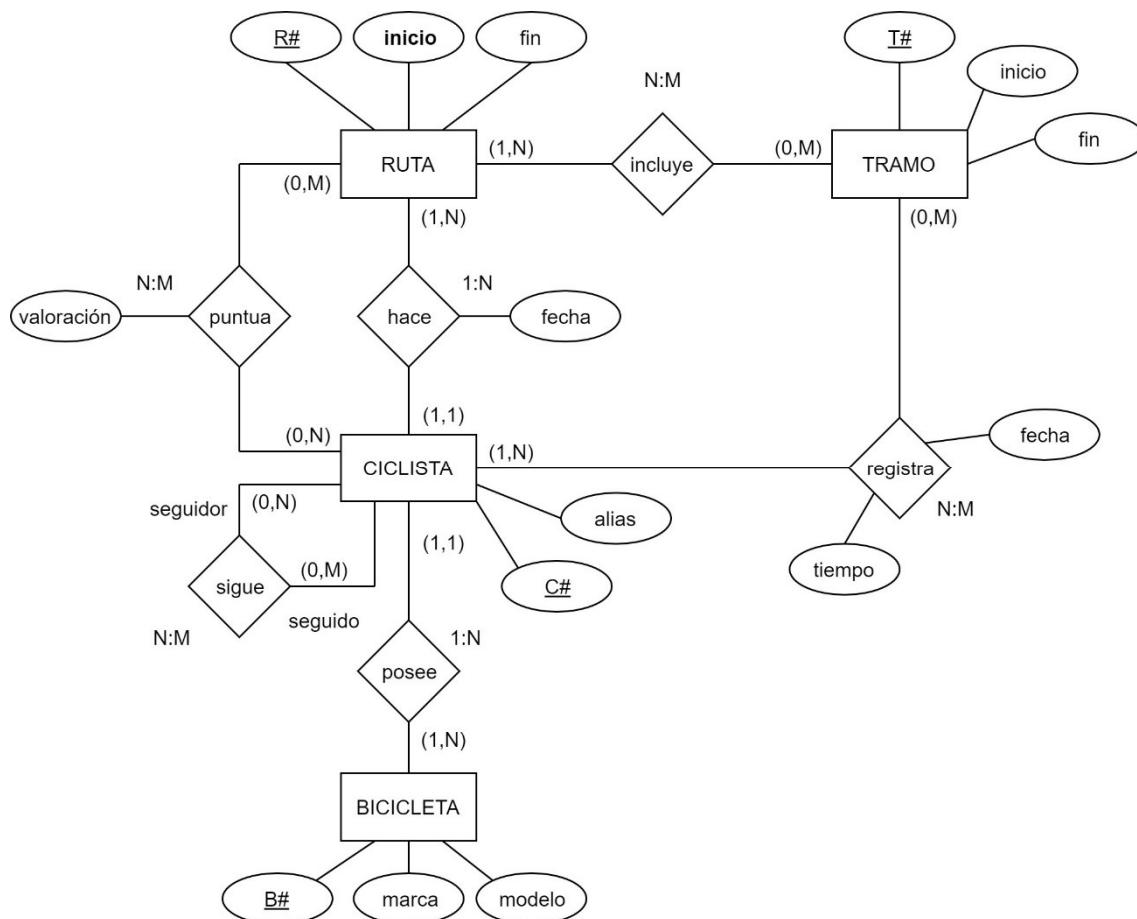
Cuando un usuario de Internet se conecta al SIG podrá realizar cualquier tipo de consulta como, por ejemplo, saber el número total de habitantes que tienen las provincias bañadas por un determinado río. Además, debe existir un administrador de la aplicación que se encargará de dar altas y bajas, realizar modificaciones y una copia de seguridad mensual en la base de datos.

Se pide realizar el Modelo Entidad-Relación del problema descrito anteriormente empleando la notación de Chen.



 ETS I SISTEMAS INFORMÁTICOS	Bases de Datos. Curso 2019 / 2020		09 / 01 / 2020
	Apellidos: _____		Nombre: _____
	DNI: _____	Núm. matrícula: _____	Grupo: _____

Ejercicio 2 (1,5 puntos). Una *startup* ha diseñado el siguiente Modelo Entidad-Relación para el desarrollo de su red social de ciclistas aficionados:



Se pide transformar este diagrama en un Modelo Relacional mediante las reglas del “Paso a Tablas”.

CICLISTA (C#, alias)

RUTA (R#, inicio, fin, C#, fecha)

TRAMO (T#, inicio, fin)

BICICLETA (B#, marca, modelo, C#)

SIGUE (C#seguidor, C#seguido)

PUNTUA (C#, R#, valoración)

INCLUYE (R#, T#)

REGISTRA (C#, T#, fecha, tiempo)

Ejercicio 3 (2,0 puntos). Dada el siguiente Modelo Relacional en el que los atributos subrayados representan las claves primarias de las relaciones y los atributos en cursiva representan las claves foráneas:

ALUMNOS (A#, nombre, apellido)

PRACTICAS (P#, tema, curso)

REALIZA (A#, P#, nota)

Se pide dar respuesta en Álgebra Relacional a cada una de las consultas planteadas a continuación:

1. Obtener el nombre de los alumnos que no han aprobado ninguna práctica de las que han realizado.


$$\Pi_{\text{nombre}} [\Pi_{A\#}(\text{Realiza}) - \Pi_{A\#}(\sigma_{\text{nota} \geq 5}(\text{Realiza}))] * \text{Alumnos}]$$

2. Obtener el código (P#) de las prácticas de primer curso que han sido aprobadas por todos los alumnos.

$$\Pi_{P\#,A\#} [\sigma_{\text{nota} \geq 5}(\text{Realiza}) * \sigma_{\text{curso} = 1}(\text{Practicas})] \div \Pi_{A\#}(\text{Alumnos})$$

3. Obtener el nombre de los alumnos que, habiendo sacado al menos un 9 en alguna práctica, han suspendido la práctica de código 'P03'.

$$\Pi_{\text{nombre}} [[\Pi_{A\#}(\sigma_{\text{nota} \geq 9}(\text{Realiza})) \cap \Pi_{A\#}(\sigma_{P\#='P03' \text{ AND } \text{nota} \leq 5}(\text{Realiza}))] * \text{Alumnos}]$$

 ETSI SISTEMAS INFORMÁTICOS	Bases de Datos. Curso 2019 / 2020		09 / 01 / 2020
	Apellidos: _____		Nombre: _____
	DNI: _____	Núm. matrícula: _____	Grupo: _____

Ejercicio 4 (2,0 puntos). Una conocida red social desea incorporar nuevas funcionalidades para sus usuarios que requieren la codificación de algunas consultas en lenguaje SQL. Sabiendo que su base de datos ha sido creada mediante las siguientes sentencias SQL:

```
CREATE TABLE Usuarios (
    id INTEGER UNIQUE NOT NULL,
    nombre VARCHAR(250) NOT NULL,
    apellidos VARCHAR(250) NOT NULL,
    fechaNacimiento DATE NOT NULL,
    PRIMARY KEY(ID)
);
```

```
CREATE TABLE Publicaciones (
    id INTEGER UNIQUE NOT NULL,
    idUsuario INTEGER UNIQUE NOT NULL,
    titulo VARCHAR(250) NOT NULL,
    mensaje VARCHAR(3000) NOT NULL,
    PRIMARY KEY(ID),
    CONSTRAINT FOREIGN KEY (idUsuario)
        REFERENCES Usuarios(id)
);
```

```
CREATE TABLE Comentan (
    idPublicación INTEGER NOT NULL,
    idUsuario INTEGER NOT NULL,
    fecha DATE NOT NULL,
    mensaje VARCHAR(3000) NOT NULL,
    PRIMARY KEY (idPublicación, idUsuario, fecha),
    CONSTRAINT FOREIGN KEY (idPublicación)
        REFERENCES Publicaciones(id),
    CONSTRAINT FOREIGN KEY (idUsuario)
        REFERENCES Usuarios(id)
);
```

Se pide dar respuesta en lenguaje SQL a cada una de las consultas planteadas a continuación:

1. Obtener todos los datos de aquellos usuarios que no han publicado nunca pero que sí han comentado alguna publicación.


```
SELECT Usuarios.*
FROM Usuarios
WHERE id NOT IN (SELECT idUsuario FROM Publicaciones)
      AND id IN (SELECT idUsuario FROM Comentan);
```

2. Obtener el *nombre* de usuario, el *id* de la publicación y el total de comentarios realizados por el usuario en la publicación para aquellas publicaciones que han sido comentadas más de 5 veces por el usuario.

```
SELECT nombre, idPublicación, total
FROM Usuarios
    INNER JOIN (SELECT idUsuario, idPublicación, COUNT(*) as total
                FROM Comentan
                GROUP BY idUsuario, idPublicación
                HAVING COUNT(*) > 5) Subconsulta
ON Usuarios.id = Subconsulta.idUsuario;
```

3. Obtener el *id* y el *título* de aquellas publicaciones que hayan sido comentadas por todos los usuarios.

[illegible]

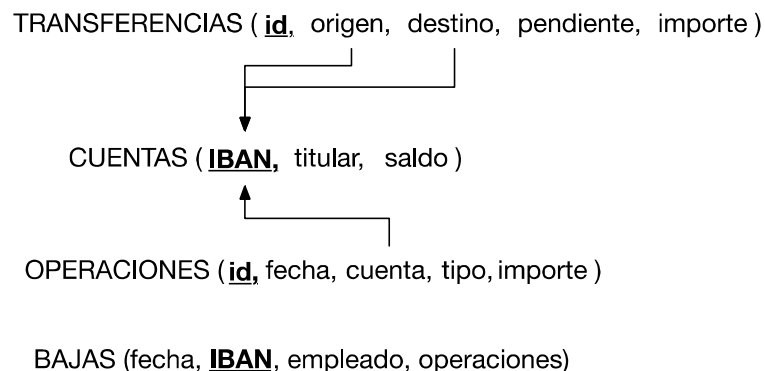
	Bases de Datos. Curso 2019 / 2020		09 / 01 / 2020
	Apellidos: _____	Nombre: _____	
	DNI: _____	Núm. matrícula: _____	Grupo: _____

Ejercicio 5 (1,5 punto). El Banco Revilla está actualizando su sistema de gestión, en concreto la operativa de la cancelación de cuentas. A la directora le han hablado sobre las bondades del código almacenado de MySQL, por lo que desea que su base de datos esté dotada con la siguiente operativa:

“Cuando se dé de baja una cuenta en nuestros sistemas, será necesario que se realicen de forma automática las siguientes acciones:

1. *Comprobar que el importe total de las transferencias (tabla **TRANSFERENCIAS**) cuyo origen es la cuenta que se va a dar de baja **NO SUPERA** el saldo de esta (tabla **CUENTAS**). En el caso de que el importe sea superior al saldo, la legislación nos obliga a cancelar la baja de la cuenta, mostrando un mensaje de error para el empleado (no se realizarán el resto de las acciones que aparecen a continuación).*
2. *Añadir una fila al registro de cuentas dadas de baja (tabla **BAJAS**), reflejando la fecha y hora en la cual se ha dado de baja la cuenta, su IBAN, el empleado del banco que ha tramitado la baja (usuario actual de la base de datos) y el número total de operaciones registradas en dicha cuenta (tabla **OPERACIONES**).”*

El Modelo Relacional utilizado por la base de datos MySQL, que ha sido implantada como sistema de almacenaje de información persistente por el Banco Revilla, es el siguiente:



Se pide, implementar un Trigger para MySQL que permita realizar la operativa descrita anteriormente cuando se da de baja una cuenta bancaria del Banco Revilla.

```

delimiter //
CREATE TRIGGER baja_cuentas BEFORE DELETE ON CUENTAS
FOR EACH ROW
BEGIN
    DECLARE transf, disponible DECIMAL(10,2);
    DECLARE opers INTEGER;

    SELECT SUM(importe) INTO transf
    FROM TRANSFERENCIAS
    WHERE origen = OLD.IBAN;


    SELECT saldo INTO disponible
    FROM CUENTAS
    WHERE IBAN = OLD.IBAN;

    IF transf > saldo THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Las transferencias superan al saldo de la
cuenta';
    END IF;

    SELECT count(*) INTO opers
    FROM OPERACIONES
    WHERE cuenta = OLD.IBAN;

    INSERT INTO BAJAS (fecha, IBAN, empleado, operaciones)
    VALUES (NOW(), OLD.IBAN, USER(), opers);
END//
delimiter ;

```


	Bases de Datos. Curso 2019 / 2020		09 / 01 / 2020
	Apellidos: _____		Nombre: _____
	DNI: _____	Núm. matrícula: _____	Grupo: _____

Ejercicio 6 (1,0 punto). Explica en qué consisten los formatos de fichero CSV, XML y JSON, cuáles son sus respectivas ventajas e inconvenientes, un ejemplo de cada uno de ellos y un caso de uso para cada formato.

CSV

Los ficheros CSV almacenan información estructurada de forma que en cada línea se almacena un registro, mientras que los elementos de cada registro se separan por un carácter delimitador. El orden de estos elementos se mantienen a lo largo del fichero.

Las ventajas de usar ficheros CSV son:

1. Al ser información estructurada, su lectura y escritura es rápida y sencilla.
2. Salvo por el separador, no añade información adicional o superflua a los datos.
3. Tiene un formato fácilmente entendible y editable.
4. Está considerado como un formato estándar (RFC4180).
5. Es fácil de generar.

Inconvenientes

1. Solo permite tipos de datos simples.
2. Hay que controlar que el separador no aparezca en los datos.
3. Hay que utilizar caracteres de escape para representar cadenas de texto que incluyan símbolos especiales.
4. Todos los registros deben tener el mismo número de elementos.

El formato CSV se utiliza principalmente en las siguientes situaciones:

- Dada su estructura secuencial intrínseca, es muy común usar CSV para almacenar datos de procesos temporales (p.e. sensores), ya que es fácil escribir al final de los ficheros.
- Análisis de datos, donde se realizan operaciones sobre el conjunto de datos completo.

Un ejemplo:

```
longitude,latitude,time,station,wmo_platform_code,T_25
0.0,0.0,2015-05-23T12:00:00Z,0n0e,13010,27.89
0.0,0.0,2015-05-24T12:00:00Z,0n0e,13010,27.5
```

XML

El objeto básico de XML es el documento. Para construir un documento XML contamos con los siguientes conceptos:

- Elementos, que se identifican por su etiqueta de inicio y su etiqueta final y que pueden estar formado a su vez por otros elementos.
- Atributos, que se usan para describir propiedades y características de los elementos a los que se añaden y se incluyen en la etiqueta inicial a continuación del nombre de la misma.

Ventajas:

- Metalenguaje: permite definir lenguajes para representar información.
- Simplicidad: facilidad de procesamiento por software y de entendimiento por personas
- Utilizable con cualquier lenguaje o alfabeto (representa el estándar unicode).
- Sensitivo a mayúsculas y minúsculas.
- Gramática de obligado cumplimiento.
- Auto-descriptivo: datos como texto, metadatos como etiquetas y atributos.
- Separa:
 - Estructura (metadatos): DTD, Xml-Schema
 - Contenido (datos): documento xml
 - Apariencia (presentación): XSL, CSS
- Es un estándar para intercambio de datos en la Web y aplicaciones en general
- Poderosas técnicas para búsqueda de información: Xpath, Xquery
- APIs en programación: DOM y SAX

Inconvenientes:

- Las etiquetas dificultan la legibilidad de los ficheros además de incrementar el tamaño de los mismos.

Ejemplo:

```
<empleado>


    <nombre>Pepe</nombre>

    <edad>47</edad>

</empleado>
```

JSON

- Formato de datos semi-estructurados.
- Es una representación textual de objetos de datos.
- Permite el intercambio sencillo de información entre servicios.
- Representa objetos usando pares atributo-valor.
- Formato para SGBD no relacionales (NoSQL) como _MongoDB_.
- Su sintaxis es un subconjunto de JavaScript.

 Universidad Politécnica de Madrid	Bases de Datos. Curso 2019 / 2020		09 / 01 / 2020
	Apellidos: _____		Nombre: _____
	DNI: _____	Núm. matrícula: _____	Grupo: _____

Ventajas:

- Una de las principales ventajas de JSON es que es auto-descriptivo y fácil de entender
- Además, es más compacto que XML ya que elimina las etiquetas.

Inconvenientes:

- Representar relaciones N:M no es fácil.

Ejemplo:

```
{
  "employees": [
    { "firstName": "John", "lastName": "Doe" },
    { "firstName": "Anna", "lastName": "Smith" },
    { "firstName": "Peter", "lastName": "Jones" }
  ]
}
```