**EXAMEN FINAL DE LA 1ª EVALUACIÓN:**

1. **Dado el siguiente enunciado diseña el modelo E/R correspondiente. (3 Puntos):**

Tenemos que diseñar una base de datos para gestionar las reservas de hoteles y vuelos que realizan los clientes de una agencia de viajes:

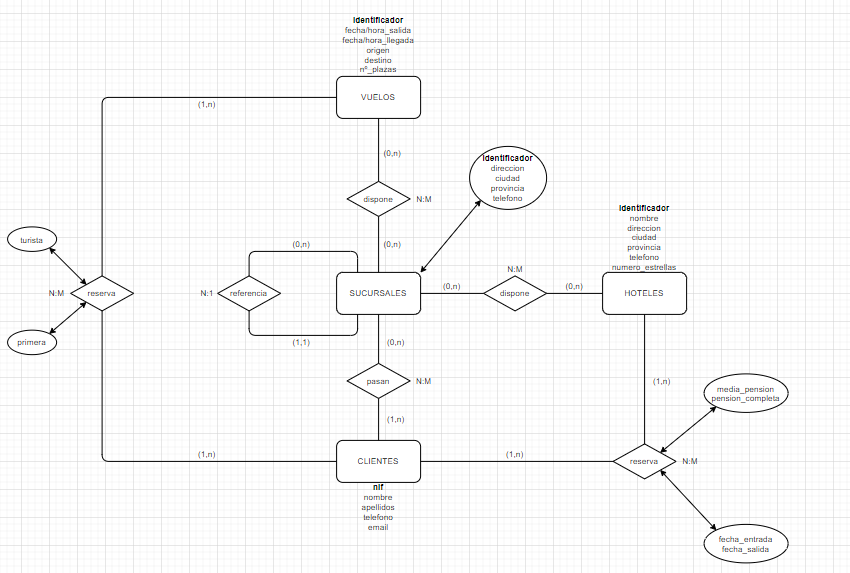
La agencia de viajes tiene oficinas **sucursales**. Cada sucursal está definida por el **identificador de sucursal**, dirección, ciudad, provincia y teléfono. Una sucursal tiene siempre otra oficina de referencia con la que se comunica normalmente.

Por su parte, los clientes normalmente se pasan por las sucursales para contratar vuelos y estancias en hoteles, sabiendo que:

* Cada **vuelo** está definido por un **identificador**, fecha/hora de salida, fecha/hora de llegada, origen, destino y número de plazas totales.
* Cada **hotel** está definido por el **identificador del hotel**, nombre, dirección, ciudad, provincia, teléfono y número de estrellas.
* La información que se desea almacenar para cada **cliente** es un **nif**, nombre, apellidos, teléfono y email.

También habrá que tener en cuenta la siguiente información:

* A la agencia de viajes le interesa conocer a través de qué sucursal ha contratado cada cliente los servicios de vuelo y alojamiento.
* A la hora de reservar un vuelo el cliente puede elegir cualquiera de los vuelos que ofrece la agencia y en qué clase (turista o primera) desea viajar.
* El cliente se puede hospedar en cualquiera de los hoteles que ofrece la agencia, y elegir el régimen de hospedaje (media pensión o pensión completa). Siendo significativa la fecha de entrada y de salida.



En el caso de “turista”, “primera”, “media\_pension”, “pension\_completa”; se tratan de campos de tipo BOOLEAN.

1. **Dado el siguiente el modelo E/R anterior, pásalo al modelo relacional. (1,5 Puntos):**

VUELOS (identificador, fecha/hora\_salida, fecha/hora\_llegada, origen, destino, nº\_plazas).

HOTELES (identificador, nombre, dirección, ciudad, provincia, teléfono, numero\_estrellas).

SUCURSALES (identificador, dirección, ciudad, provincia, teléfono, identificador\_SUCURSALES).

CLIENTES (nif, nombre, apellidos, teléfono, email).

RESERVA\_HOTEL (nif\_CLIENTES, identificador\_HOTELES, media\_pension, pension\_completa, fecha\_entrada, fecha\_salida).

RESERVA\_VUELOS (nif\_CLIENTES, identificador\_VUELOS, turista, primera).

DISPOSICION\_VUELOS (identificador\_SUCURSALES, identificador\_VUELOS).

DISPOSICION\_HOTELES (identificador\_SUCURSALES, identificador\_HOTELES).

CONTRATO\_SUCURSALES (identificador\_SUCURSALES, nif\_CLIENTES).

1. **Ejercicio de Normalización. (1,5 Puntos):**

Teniendo en cuenta que:

* Una editorial tiene una sede y una persona de contacto (aunque no se encuentran registradas actualmente, pero se registrará en un futuro)
* Se quiere también añadir en un futuro el año de edición del libro.

Realiza el modelo relacional de modo que el diseño se encuentre al menos en 3ª forma normal.

LIBROS (Cod\_libro, ISBN, Titulo, Autor, año\_edicion).

AUTORES (Autor, Pais\_autor, ISBN).

EDITORIALES (Editorial, ISBN).

PRESTAMO (ISBN, Tele, Fecha\_prestamo, Fecha\_devolucion).

USUARIO (Apellido1, Apellido2, Nombre, Tele).

PERSONA\_CONTACTO (nombre, apellidos, teléfono, nif).

1. **Dada la base de datos empleado (script2.sql), realiza las siguientes consultas (4 puntos):**
2. Lista el código de los departamentos de los empleados que aparecen en la tabla empleado, eliminando los códigos que aparecen repetidos:

**SELECT** **DISTINCT**(codigo)

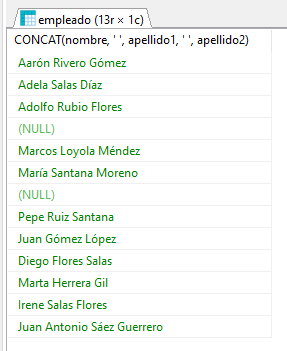
**FROM** empleado;



1. Lista el nombre y apellidos de los empleados en una única columna:

**SELECT** **CONCAT**(nombre, ' ', apellido1, ' ', apellido2)

**FROM** empleado;

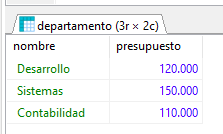


1. Devuelve una lista con el nombre de los departamentos y el presupesto, de aquellos que tienen un presupuesto entre 100000 y 200000 euros:

**SELECT** nombre, presupuesto

**FROM** departamento

**WHERE** presupuesto **BETWEEN** 100000 **AND** 200000;



1. Devuelve un listado con los empleados y los datos de los departamentos donde trabaja cada uno. Ordena el resultado, en primer lugar, por el nombre del departamento (en orden alfabético) y en segundo lugar por los apellidos y el nombre de los empleados:

**SELECT** empleado.\*, departamento.\*

**FROM** empleado

**INNER** **JOIN** departamento **ON** empleado.codigo\_departamento=departamento.codigo

**ORDER** **BY** departamento.nombre **ASC**, empleado.apellido1, empleado.apellido2, empleado.nombre;



1. Devuelve un listado con los datos de los empleados que trabajan en el departamento de Sistemas, Contabilidad o I+D. Ordena el resultado alfabéticamente:

**SELECT** empleado.\*, departamento.nombre

**FROM** empleado

**INNER** **JOIN** departamento **ON** empleado.codigo\_departamento=departamento.codigo

**WHERE** departamento.nombre **IN** ('Sistemas', 'Contabilidad', 'I+D')

**ORDER** **BY** empleado.nombre;



1. Calcula el número de empleados que trabajan en cada uno de los departamentos que tienen un presupuesto mayor a 200000 euros. Detrás del número de empleados tiene que aparecer la palabra “empleados”:

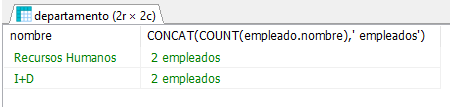
**SELECT** departamento.nombre, **CONCAT**(**COUNT**(empleado.nombre),' empleados')

**FROM** empleado

**INNER** **JOIN** departamento **ON** empleado.codigo\_departamento=departamento.codigo

**WHERE** departamento.presupuesto>200000

**GROUP** **BY** codigo\_departamento;



1. Devuelve el nombre del departamento con menor presupuesto y la cantidad 3 que tiene asignada (sin usar limit):

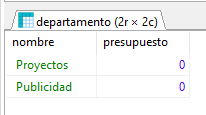
**SELECT** departamento.nombre, departamento.presupuesto

**FROM** departamento

**WHERE** departamento.presupuesto=(

**SELECT** **MIN**(departamento.presupuesto)

**FROM** departamento);



1. Listado de los empleados que tienen el segundo apellido con más letras que el 2º apellido de Aaron:

**SELECT** empleado.nombre

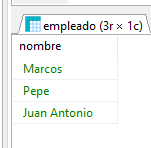
**FROM** empleado

**WHERE** **LENGTH**(empleado.apellido2)>(

**SELECT** **LENGTH**(empleado.apellido2)

**FROM** empleado

**WHERE** empleado.nombre='Aaron');



1. Lista el nombre y apellidos de los empleados en una única columna, convirtiendo todos los caracteres en mayúscula. Prueba con la función upper():

**SELECT** **UPPER**(**CONCAT**(nombre, ' ', apellido1, ' ', apellido2))

**FROM** empleado;



1. Devuelve una lista con el nombre de los empleados que tienen los departamentos que no tienen un presupuesto entre 100000 y 200000 euros:

**SELECT** empleado.nombre, departamento.presupuesto

**FROM** empleado

**INNER** **JOIN** departamento **ON** empleado.codigo\_departamento=departamento.codigo

**WHERE** departamento.presupuesto **NOT** **BETWEEN** 100000 **AND** 200000;

