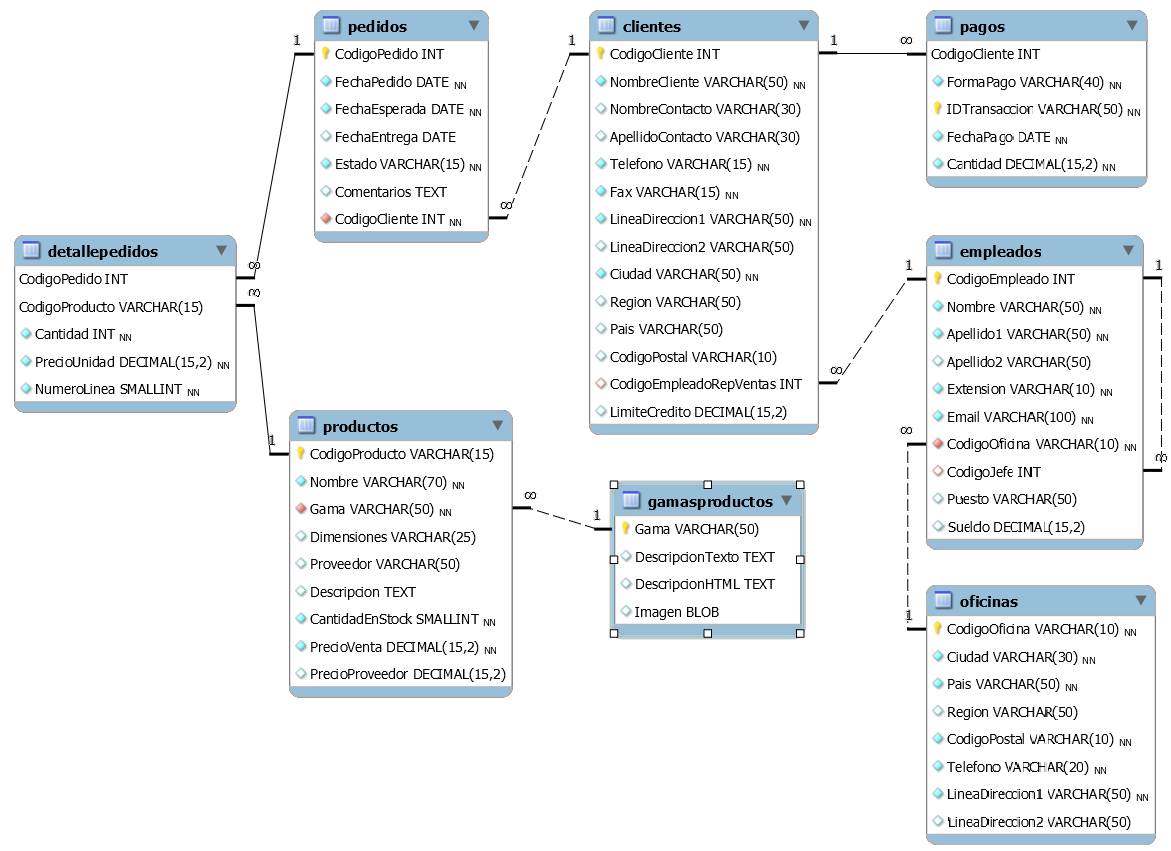
# Estructura y datos BDs

* Las inserciones de los datos se entregan en un archivo SQL aparte.

**Diagrama relacional**



# Ejercicios

**Consejo**: Primero escribe en un script aparte las consultas necesarias y, una vez que funcionan correctamente, intégralas en los procedimientos.

**Notas importantes:**

* Debes usar la plantilla y cada uno de los procedimientos debe poder ejecutarse sin errores y debe contar con TODAS las ejecuciones (los CALLs) para cada uno de los casos que figuran en las capturas.
* Se entregará un solo fichero: la plantilla.
* Cada procedimiento debe contar con su DROP PROCEDURE IF EXISTS o DROP FUNCTION IF EXISTS.
* No debe contar con USE.
* Es decir, **queremos que al cargar vuestro fichero en MySQL WorkBench tan solo debamos ejecutar el fichero completo y aparezcan todos los resultados en el orden en el que están las capturas** (o los resultados que hayáis conseguido hacer). Si habéis incluido código que no funciona, pero queréis que cuente el código que habéis escrito comentarlo para que no de error. **Si figuran errores se penalizará**.
* Es imprescindible que cada ejercicio tenga el **código comentado**, ya que los comentarios tendrán un peso en la nota según la rúbrica.

1. (**1 punto**) Implementa una función llamada “fn\_numEmpleadosOficina” que, dado por parámetro el código de una oficina, devuelva el Nº de empleados que tiene.

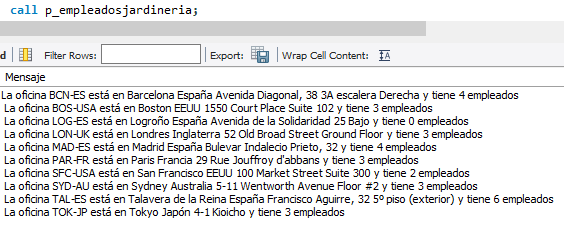
* Si no existe la oficina devolverá -1
* Si la oficina existe devolverá el Nº de empleados (Si no tiene ningún empleado deberá devolver 0).

|  |  |
| --- | --- |
|  |  |

1. (**2 puntos**) Implementa un procedimiento que utilice cursores llamado “p\_empleadosjardineria” y que no reciba parámetros.

Para cada una de las oficinas debe mostrar su ubicación (ciudad, país, región, líneas de dirección 1 y 2) y cuántos empleados trabajan en ella de la siguiente forma:

* Compruebe si existe una segunda línea de dirección, en caso contrario no debe mostrar este campo por pantalla.
* Para cada una de las oficinas debe mostrar su ubicación (ciudad, país, región, líneas de dirección 1 y 2) y cuántos empleados trabajan en ella.
* Debes usar la función fn\_numEmpleadosOficina.

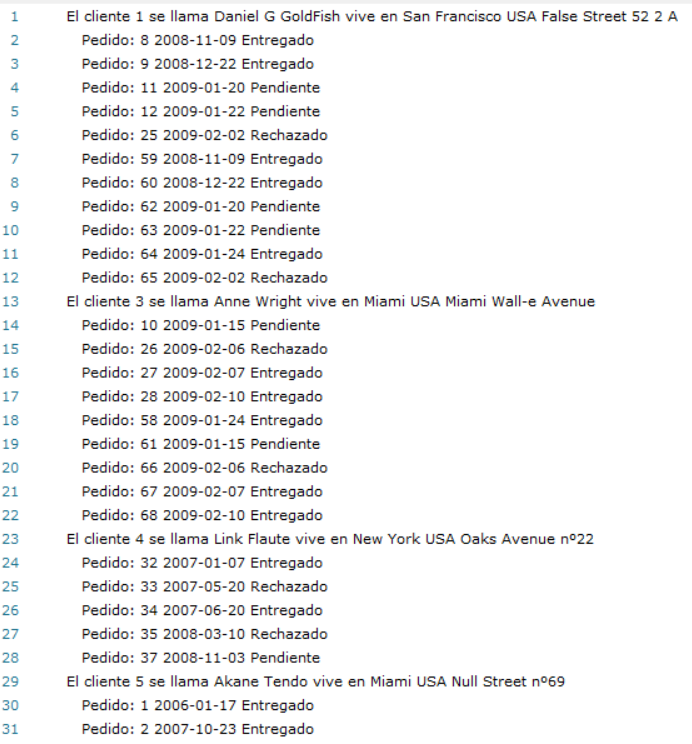


* Recuerda, el procedimiento debe mostrar un solo mensaje con todas las líneas.

1. (**3 puntos**) Implementa un procedimiento que utilice cursores anidados llamado “p\_clientesjardineria2” que no reciba parámetros. El procedimiento debe incluir la información de los pedidos de cada cliente:
   1. codigoPedido
   2. fechaPedido
   3. estado

La información debe mostrarse formateada de la siguiente manera:





Etc...

* Recuerda, el procedimiento debe mostrar un solo mensaje con todas las líneas.

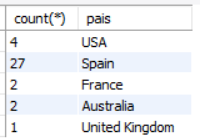
1. (**2 puntos**) Implementa un procedimiento que utilice cursores llamado “p\_cambiaPaisCliente” que no reciba parámetros. El procedimiento debe cambiar el nombre del país de los clientes cuyo país sea “España” para que dicho nombre aparezca en inglés (‘Spain’) y coincida con el formato de país de todos los demás clientes de la tabla.

* Ojo: En el cursor hay que cargar SOLO los clientes que cuyo país sea ‘España’ para luego modificar uno por uno esos países usando la clave primaria exclusivamente en el WHERE.

Una vez terminado, el resultado de seleccionar cuántos clientes hay en cada país debe ser el siguiente:







(**Opcional**) Implementa un procedimiento que utilice cursores anidados llamado “p\_clientesjardineria3” que no reciba parámetros. El procedimiento debe ampliar la información mostrada en el procedimiento p\_clientesjardineria2 (pero no debe llamar en ningún momento a “p\_clientesjardineria2”) incluyendo ahora información de los productos de los pedidos:

* 1. CodigoProducto
  2. Nombre
  3. Gama
  4. PrecioVenta

La información debe mostrarse formateada de la siguiente manera:





*... más clientes...*

* Recuerda, el procedimiento debe mostrar un solo mensaje con todas las líneas.