Práctica 3. Consultas SQL

1. Enunciado

En este ejercicio se realizarán consultas SQL que respondan a las preguntas que se plantearán sin utilizar QBE. Dada una base de datos denominada Empresa y definida por las siguientes relaciones:

Empleados(DNI: 9, Sueldo: Entero largo)

Domicilios(DNI: 9, Calle: 50, Código postal: 5)

Teléfonos(DNI: 9, Teléfono: 9)

Códigos postales(Código postal: 5, Población: 50, Provincia: 50)

donde la tabla Empleados almacena los empleados de una empresa, la tabla Domicilios almacena los domicilios de estos empleados, la tabla Teléfonos almacena los teléfonos de los empleados y la tabla Códigos postales almacena los códigos postales de España con referencia a la población y provincia correspondientes.

Hay que considerar que un empleado puede tener varios domicilios o que puede compartir vivienda con otro empleado e incluso que puede no conocerse su domicilio, y también que puede tener varios (o ningún) teléfonos, así como un teléfono puede estar compartido por varios empleados.

A continuación se pide resolver en SQL una serie de consultas. Estas consultas se deben denominar XXY, donde XX es el número del apartado (con un cero a la izquierda si es menor que 10) e Y la letra del subapartado. Por ejemplo: 03b es el nombre de la consulta del subapartado b) del apartado 3.

1) Realizar el diseño físico, determinando las claves, restricciones de cardinalidad y de integridad referencial. (Usar SQL al menos para la creación de las tablas. Las restricciones de integridad referencial se pueden crear en QBE.)
2) Poblar las tablas con los datos que aparecen al final.
3) Listado de empleados que muestre Nombre, Calle y Código postal ordenados por Código postal y Nombre de dos formas diferentes:
   a) Con SELECT.
   b) Con JOIN.
4) Listado de los empleados ordenados por nombre que muestre Nombre, DNI, Calle, Código postal, Teléfono de dos formas diferentes:
   a) Sólo los empleados que tengan teléfono.
   b) Los empleados que tengan teléfono como los que no.

5) Listado de los empleados que muestre Nombre, DNI, Calle, Población, Provincia y Código postal ordenados por nombre.
6) Listado de los empleados que muestre Nombre, DNI, Calle, Población, Provincia, Código postal y Teléfono ordenados por nombre.
7) Incrementar en un 10% el sueldo de todos los empleados, de forma que el sueldo aumentado no supere en ningún caso 1.900 €.
8) Deshacer la operación anterior con una consulta (comprobar que los datos coinciden con los de la tabla original).
9) Repetir los dos pasos anteriores con el límite 1.600 €.
10) Listado del número total de empleados, el sueldo máximo, el mínimo y el medio.
11) Listado de sueldo medio y número de empleados por población ordenado por población.
12) Listado de provincias con códigos postales ordenado por población. En la cabecera de las columnas deben aparecer las provincias y en cada columna los códigos postales de las localidades de cada provincia, de la forma:

Población	Barcelona	Córdoba	Madrid	Zaragoza
Arganda			28040	
Lucena		14900		
Madrid	28040			
Madrid	28000			
...				

13) Agregar restricciones de integridad de dominio con las reglas de validación (el sueldo debe estar comprendido entre 0 y 120.000 €.)
14) Agregar máscaras de entrada para todos los campos que lo acepten.
15) Agregar formato de salida para todos los campos que lo acepten.

Tabla **Empleados**

Nombre	DNI	Sueldo
Antonio Arjona	12345678A	5.000 €
Carlota Cerezo	12345678C	1.000 €
Laura López	12345678L	1.500 €
Pedro Pérez	12345678P	2.000 €

Tabla **Códigos postales**

Código postal	Población	Provincia
08050	Parets	Barcelona
14200	Peñarroya	Córdoba
14900	Lucena	Córdoba
28040	Madrid	Madrid
50008	Zaragoza	Zaragoza
28004	Arganda	Madrid
28000	Madrid	Madrid

Tabla **Teléfonos**

DNI	Teléfono
12345678C	611111111

12345678C	931111111
12345678L	913333333
12345678P	913333333
12345678P	644444444

Tabla **Domicilios**

DNI	Calle	Código postal
12345678A	Avda. Complutense	28040
12345678A	Cántaro	28004
12345678P	Diamante	15200
12345678P	Carbón	14900
12345678L	Diamante	14200

Solución:

1) Realizar el diseño físico, determinando las claves y restricciones de unicidad, existencia e integridad referencial. (Usar SQL al menos para la creación de las tablas. Las restricciones de integridad referencial se pueden crear en QBE.)

a) Tabla Empleados:

Restricciones:

- Clave: {DNI}
- Unicidad: Ninguna.
- Existencia: {Nombre}
- Integridad referencial: Ninguna.

```
CREATE TABLE Empleados(Nombre TEXT(50) NOT NULL, DNI TEXT(9) PRIMARY
KEY, SUELDO INTEGER)
```

b) Tabla Códigos postales:

Restricciones:

- Clave: {Código postal}
- Unicidad: Ninguna.
- Existencia: {Población}, {Provincia}
- Integridad referencial: Ninguna.

```
CREATE TABLE [Códigos postales]([Código postal] TEXT(5) PRIMARY KEY,
Población TEXT(50) NOT NULL, Provincia TEXT(50) NOT NULL)
```

c) Tabla Teléfonos:

Restricciones:

- Clave: {DNI, Teléfono}
- Unicidad: Ninguna.
- Existencia: {Población}, {Provincia}
- Integridad referencial: {Teléfonos.DNI} $\subseteq$ {Empleados.DNI}.

```
CREATE TABLE Teléfonos(DNI TEXT(9) REFERENCES Empleados(DNI), Teléfono
TEXT(9) NOT NULL, PRIMARY KEY (DNI, Teléfono))
```

d) Tabla Domicilios:

Restricciones:

- Clave: {DNI, Calle, [Código postal]}
- Unicidad: Ninguna.
- Existencia: Ninguna
- Integridad referencial: {Domicilios.DNI} ⊆ {Empleados.DNI}, {Domicilios.Código postal } ⊆ {Códigos postales.Código postal }..

```
CREATE TABLE Domicilios(DNI TEXT(9) REFERENCES Empleados(DNI), Calle
TEXT(50), [Código postal] TEXT(5) REFERENCES[Códigos postales]([Código
postal]), PRIMARY KEY (DNI, Calle, [Código postal]))
```

El resultado de las relaciones mostrando las restricciones de cardinalidad es:

3) Listado de empleados que muestre Nombre, Calle y Código postal ordenados por Código postal y DNI de dos formas diferentes:

   a) Con SELECT.

```
SELECT Nombre, Calle, [Código postal]
FROM Empleados, Domicilios
WHERE Empleados.DNI=Domicilios.DNI
ORDER BY [Código postal], Nombre;
```

   b) Con INNER JOIN:

```
SELECT Nombre, Calle, [Código postal]
FROM Empleados INNER JOIN Domicilios
ON Empleados.DNI=Domicilios.DNI
ORDER BY [Código postal], Nombre;
```

Resultado (el mismo para ambas consultas):

Nombre	Calle	Código postal
Antonio Arjona	Avda.	28040
Antonio Arjona	Cántaro	28004
Laura López	Diamante	14200
Pedro Pérez	Diamante	14200
Pedro Pérez	Carbón	14900

4) Listado de los empleados ordenados por nombre que muestre Nombre, DNI, Calle, Código postal, Teléfono de dos formas diferentes:

   a) Sólo los empleados que tengan teléfono.

```
SELECT Nombre, Empleados.DNI, Calle, [Código postal], Teléfono
FROM
(Empleados LEFT JOIN Domicilios ON Empleados.DNI=Domicilios.DNI) INNER
JOIN Teléfonos ON Teléfonos.DNI=Empleados.DNI
ORDER BY Nombre;
```

Nombre	DNI	Calle	Código postal	Teléfono
Carlota Cerezo	12345678C			931111111
Carlota Cerezo	12345678C			611111111
Laura López	12345678L	Diamante	14200	913333333
Pedro Pérez	12345678P	Carbón	14900	644444444
Pedro Pérez	12345678P	Carbón	14900	913333333
Pedro Pérez	12345678P	Diamante	14200	644444444
Pedro Pérez	12345678P	Diamante	14200	913333333

   b) Los empleados que tengan teléfono como los que no.

```
SELECT Nombre, Empleados.DNI, Calle, [Código postal], Teléfono
FROM Empleados LEFT JOIN (Teléfonos LEFT JOIN Domicilios ON
Teléfonos.DNI=Domicilios.DNI) ON Empleados.DNI=Teléfonos.DNI
ORDER BY Nombre;
```

Nombre	DNI	Calle	Código postal	Teléfono
Antonio Arjona	12345678A			
Carlota Cerezo	12345678C			931111111
Carlota Cerezo	12345678C			611111111
Laura López	12345678L	Diamante	14200	913333333
Pedro Pérez	12345678P	Carbón	14900	644444444
Pedro Pérez	12345678P	Diamante	14200	644444444
Pedro Pérez	12345678P	Carbón	14900	913333333
Pedro Pérez	12345678P	Diamante	14200	913333333

5) Listado de los empleados que muestre Nombre, DNI, Calle, Población, Provincia y Código postal ordenados por nombre.

```
SELECT Nombre, Empleados.DNI, Calle, Población, Provincia,
Domicilios.[Código postal]
FROM (Empleados LEFT JOIN Domicilios ON Empleados.DNI=Domicilios.DNI)
LEFT JOIN [Códigos postales] ON Domicilios.[Código postal]=[Códigos
postales].[Código postal]
ORDER BY Nombre;
```

Nombre	DNI	Calle	Población	Provincia	Código postal
Antonio Arjona	12345678A	Cántaro	Arganda	Madrid	28004
Antonio Arjona	12345678A	Avda.	Madrid	Madrid	28040
Carlota Cerezo	12345678C				
Laura López	12345678L	Diamante	Peñarroya	Córdoba	14200
Pedro Pérez	12345678P	Carbón	Lucena	Córdoba	14900
Pedro Pérez	12345678P	Diamante	Peñarroya	Córdoba	14200

6) Listado de los empleados que muestre Nombre, DNI, Calle, Población, Provincia, Código postal y Teléfono ordenados por nombre.

```
SELECT Nombre, Empleados.DNI, Calle, Población, Provincia, [Códigos
postales].[Código postal], Teléfono
FROM
((Empleados LEFT JOIN Teléfonos ON Empleados.DNI=Teléfonos.DNI)
  LEFT  JOIN Domicilios ON Empleados.DNI=Domicilios.DNI)
LEFT JOIN [Códigos postales]
ON Domicilios.[Código postal]=[Códigos postales].[Código postal]
ORDER BY Nombre;
```

Nota: La cláusula LEFT JOIN más externa (la que aparece en último lugar) no tendría que ser necesariamente una reunión externa, bastaría con una interna. Es necesario hacer uso de ella porque Access no permite anidar reuniones INNER JOIN dentro de reuniones LEFT o RIGHT JOIN.

Nombre	DNI	Calle	Población	Provincia	Código postal	Teléfono
Antonio Arjona	12345678A	Cántaro	Arganda	Madrid	28004	
Antonio Arjona	12345678A	Avda.	Madrid	Madrid	28040	
Carlota Cerezo	12345678C					931111111
Carlota Cerezo	12345678C					611111111
Laura López	12345678L	Diamante	Peñarroya	Córdoba	14200	913333333
Pedro Pérez	12345678P	Carbón	Lucena	Córdoba	14900	644444444
Pedro Pérez	12345678P	Carbón	Lucena	Córdoba	14900	913333333
Pedro Pérez	12345678P	Diamante	Peñarroya	Córdoba	14200	644444444
Pedro Pérez	12345678P	Diamante	Peñarroya	Córdoba	14200	913333333

7) Incrementar en un 10% el sueldo de todos los empleados, de forma que el sueldo aumentado no supere en ningún caso 1.900 €.

Una primera solución en que se puede pensar es:

```
UPDATE Empleados SET Sueldo = Sueldo*1.1
WHERE Sueldo <= 1900/1.1;
```

El resultado es, donde se resaltan los cambios:

Nombre	DNI	SUELDO
Antonio Arjona	12345678A	5000

Carlota Cerezo	12345678C	***1100***
Laura López	12345678L	***1650***
Pedro Pérez	12345678P	2000

8) Deshacer la operación anterior con una consulta (comprobar que los datos coinciden con los de la tabla original).

Una primera solución en que se puede pensar es:

```
UPDATE Empleados SET Sueldo = Sueldo/1.1
WHERE Sueldo <= 1900;
```

El resultado es el esperado, se recuperan los valores de la tabla Empleados que habían cambiado en el apartado anterior:

Nombre	DNI	SUELDO
Antonio Arjona	12345678A	5000
Carlota Cerezo	12345678C	***1000***
Laura López	12345678L	***1500***
Pedro Pérez	12345678P	2000

9) Repetir los dos pasos anteriores con el límite 1.600 €.

Con la misma consulta que en el apartado 7:

```
UPDATE Empleados SET Sueldo = Sueldo*1.1
WHERE Sueldo*1.1 <= 1600;
```

se obtiene:

Nombre	DNI	SUELDO
Antonio Arjona	12345678A	5000
Carlota Cerezo	12345678C	1100
Laura López	12345678L	1500
Pedro Pérez	12345678P	2000

Pero al intentar recuperar la información con:

```
UPDATE Empleados SET Sueldo = Sueldo/1.1
WHERE Sueldo <= 1600;
```

se obtiene:

Nombre	DNI	SUELDO
Antonio Arjona	12345678A	5000
Carlota Cerezo	12345678C	1000
Laura López	12345678L	1364

Pedro Pérez	12345678P	2000

Se ha reducido incorrectamente el sueldo de Laura López. Esto es debido a que en la ampliación de los sueldos es imposible determinar a posteriori los sueldos que han sido modificados sin ninguna otra información que el límite del sueldo. Hay una franja de indeterminación que corresponde al intervalo [Límite/1.1, Límite]. En este caso, el intervalo es [1454,56, 1600]. Es decir, para cualquier sueldo comprendido en ese intervalo no se puede decir si era el sueldo original (que no se aumentó en el 10%), o bien resultó del aumento (sueldos originales comprendidos en el intervalo [1454,56/1,1, 1600/1,1] = [1322,31, 1454,56]).

Por tanto, para recuperar correctamente la información original, es necesario anotar cuáles fueron los sueldos modificados y deshacer la operación sólo en los registros afectados. Una posible solución es crear una tabla con la clave de Empleados que almacene la referencia de los registros modificados. Al restaurar la operación es necesario contar con esta tabla para modificar sólo los registros cuya clave se encuentre en esta tabla.

Así, primero creamos la tabla con la clave de empleados:

```
CREATE TABLE [Empleados modificados](DNI CHAR(9) PRIMARY KEY);
```

Después se crea una nueva consulta que anota las tuplas modificadas:

```
INSERT INTO [Empleados modificados]
SELECT DNI FROM Empleados WHERE Sueldo*1.1 <= 1600;
```

El resultado es:

DNI
12345678C

A continuación se puede ejecutar la consulta de aumento de sueldo y, finalmente, se recuperan sólo las tuplas modificadas:

```
UPDATE Empleados SET Sueldo = Sueldo/1.1
WHERE DNI IN (SELECT DNI FROM [Empleados modificados]);
```

El resultado es:

Nombre	DNI	SUELDO
Antonio Arjona	12345678A	5000
Carlota Cerezo	12345678C	909
Laura López	12345678L	1500
Pedro Pérez	12345678P	2000

No hay que olvidar borrar las tuplas de Empleados modificados con:

```
DELETE FROM [Empleados modificados];
```

Es obvio que hacer estos pasos manualmente provocará errores al olvidar ejecutar la consulta de inserción o la de borrado de Empleados modificados. Para solucionar este inconveniente se pueden usar macros.

10) Listado del número total de empleados, el sueldo máximo, el mínimo y el medio.

```
SELECT COUNT(*) AS Empleados, MIN(Sueldo) AS [Sueldo mínimo],
MAX(Sueldo) AS [Sueldo máximo], AVG(Sueldo) AS [Sueldo medio (AVG)],
SUM(Sueldo)/Empleados AS [Sueldo medio (SUM)]
FROM Empleados;
```

El resultado es:

Empleados	Sueldo mínimo	Sueldo máximo	Sueldo medio (AVG)	Sueldo medio (SUM)
4	909	5000	2352,25	2352,25

11) Listado de sueldo medio y número de empleados por población ordenado por población.

Un empleado puede tener vivienda en dos o más poblaciones diferentes, por lo que no se puede determinar a qué población corresponde sin más datos. Para responder a la pregunta es necesario elegir entre varias alternativas entre las que se encuentran: contabilizar el empleado tantas veces como domicilios tenga, elegir arbitrariamente un solo domicilio, hacer un reparto de los datos del empleado entre todos sus domicilios de distintas poblaciones (número de empleados(1)/número de domicilios, sueldo del empleado/número de domicilios) y hacerlos corresponder a sus poblaciones correspondientes.

Por otra parte aparecen empleados de los que se desconoce su domicilio. Se deberían agrupar estos empleados para un valor de provincia desconocido (NULL).

Se darán soluciones a las tres alternativas consideradas:

a) Contabilizar el empleado tantas veces como domicilios tenga.

```
SELECT  AVG(Sueldo) AS [Sueldo medio], Población
FROM ((Empleados NATURAL LEFT OUTER JOIN Domicilios) NATURAL LEFT
OUTER JOIN [Códigos postales])
GROUP BY Población
ORDER BY Población;
```

El resultado es:

Sueldo medio	Población
1000	
5000	Arganda
2000	Lucena
5000	Madrid
1750	Peñarroya

Aquí, Antonio Arjona contabiliza tanto en Arganda como en Madrid (el único empleado en ambas poblaciones), Carlota Cerezo contabiliza en la fila de poblaciones desconocidas (la única empleada de la que no se conoce domicilio), Laura López contabiliza en Peñarroya junto a Pedro Pérez ((1500+2000)/2=1750). Éste, a su vez, contabiliza también en Lucena (por tener dos domicilios, uno en Peñarroya y otro en Lucena).

b) Elegir arbitrariamente un solo domicilio.

Se elige el domicilio que esté antes en el orden alfabético.

```
SELECT AVG(Sueldo) AS [Sueldo medio], COUNT(*) AS [Número empleados], Pob
FROM
(
 SELECT Empleados.DNI, Sueldo, MIN(Población) AS Pob
 FROM (Empleados LEFT JOIN Domicilios ON Empleados.DNI=Domicilios.DNI) LEFT OUTER JOIN [Códigos postales] ON Domicilios.[Código postal]=[Códigos postales].[Código postal]
 GROUP BY Empleados.DNI, Sueldo
)
GROUP BY Pob ORDER BY Pob;
```

El resultado es:

Sueldo medio	Número	Pob
1000	1	
5000	1	Arganda
2000	1	Lucena
1500	1	Peñarroya

Si se elige el último domicilio en el orden alfabético (con MAX), el resultado es:

Sueldo medio	Número empleados	Pob
1000	1	
5000	1	Madrid
1750	2	Peñarroya

c) Hacer un reparto de los datos del empleado entre todos sus domicilios de distintas poblaciones.

En primer lugar se determinan los empleados, sus sueldos y poblaciones (consulta 11c1):

```
SELECT Empleados.DNI, Sueldo, Población
FROM (Empleados LEFT JOIN Domicilios ON Empleados.DNI=Domicilios.DNI)
LEFT JOIN [Códigos postales] ON Domicilios.[Código postal]=[Códigos postales].[Código postal];
```

El resultado de esta consulta es:

DNI	Sueldo	Población
12345678A	5000	Madrid
12345678A	5000	Arganda
12345678C	1000	
12345678L	1500	Peñarroya
12345678P	2000	Peñarroya
12345678P	2000	Lucena

Con la siguiente consulta (11c2) se determinan los domicilios que tiene cada empleado:

```
SELECT DNI,Count(Población) as [Número domicilios]
 FROM
 11c1
 GROUP BY DNI
;
```

Y su resultado es:

DNI	Número
12345678A	2
12345678C	0
12345678L	1
12345678P	2

Ahora se incluye esta consulta en una reunión con la consulta 11c1, dando como resultado la consulta 11c3:

```
SELECT T.DNI, Sueldo, Población, [Número domicilios]
FROM
11c1 INNER JOIN
(
SELECT DNI, Count(Población) as [Número domicilios]
 FROM
 11c1
 GROUP BY DNI
) AS T
ON [11c1].DNI = T.DNI;
```

Lo cual consigue determinar el número de domicilios por cada empleado, indicando todos sus domicilios.

DNI	Sueldo	Población	Número
12345678A	5000	Madrid	2
12345678A	5000	Arganda	2
12345678C	1000		0
12345678L	1500	Peñarroya	1
12345678P	2000	Peñarroya	2

12345678P	2000	Lucena	2

Con la siguiente consulta (11c4) se determina la parte del sueldo y del empleado que se debe contabilizar por cada población. La instrucción IIF determina si Número domicilios es cero y evita la división entre 0:

```
SELECT IIF([Número domicilios]=0,Sueldo,Sueldo/[Número domicilios]) AS
ParteSueldo, DECODE([Número domicilios],0,1,1/[Número domicilios]) AS
ParteDomicilio, Población
FROM 11c3;
```

Y su resultado:

ParteSueldo	ParteDomicili	Población
2500	0,5	Madrid
2500	0,5	Arganda
1000	1	
1500	1	Peñarroya
1000	0,5	Peñarroya
1000	0,5	Lucena

Sólo falta hacer los cálculos del sueldo medio y número de empleados por provincia, teniendo en cuenta los empleados que no tienen domicilio. Se plantea en la consulta 11c5:

```
SELECT AVG(ParteSueldo) AS [Sueldo medio], SUM(ParteDomicilio) AS
[Número empleados], Población
FROM
(
SELECT IIF([Número domicilios]=0,Sueldo,Sueldo/[Número domicilios]) AS
ParteSueldo, IIF([Número domicilios]=0,1,1/[Número domicilios]) AS
ParteDomicilio, Población
FROM 11c3
)
GROUP BY Población
ORDER BY Población;
```

El resultado es:

Sueldo medio	Número	Población
1000	1	
2500	0,5	Arganda
1000	0,5	Lucena
2500	0,5	Madrid
1250	1,5	Peñarroya

12) Listado de provincias con códigos postales ordenado por población. En la cabecera de las columnas deben aparecer las provincias y en cada

columna los códigos postales de las localidades de cada provincia, de la forma:

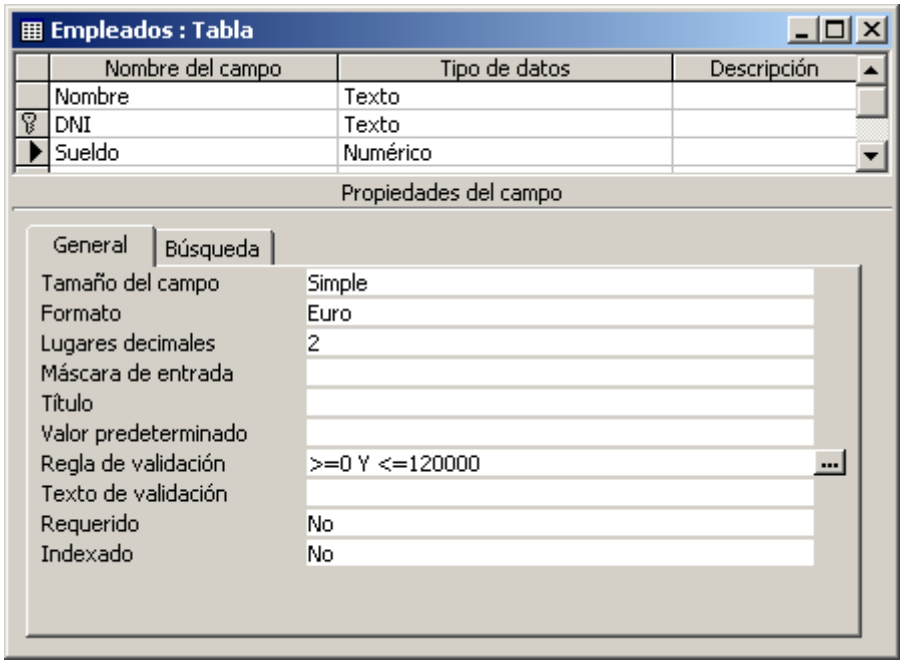
Población	Barcelona	Córdoba	Madrid	Zaragoza
Arganda			28040	
Lucena		14900		
Madrid			28040	
Madrid			28000	
...				

```
TRANSFORM [Códigos postales].[Código postal]
SELECT [Códigos postales].Población
FROM [Códigos postales]
GROUP BY [Códigos postales].Población, [Códigos postales].[Código
postal]
PIVOT [Códigos postales].Provincia;
```

El resultado es:

Población	Barcelona	Córdoba	Madrid	Zaragoza
Arganda			28004	
Lucena		14900		
Madrid			28000	
Madrid			28040	
Parets	8050			
Peñarroya		14200		
Zaragoza				50008

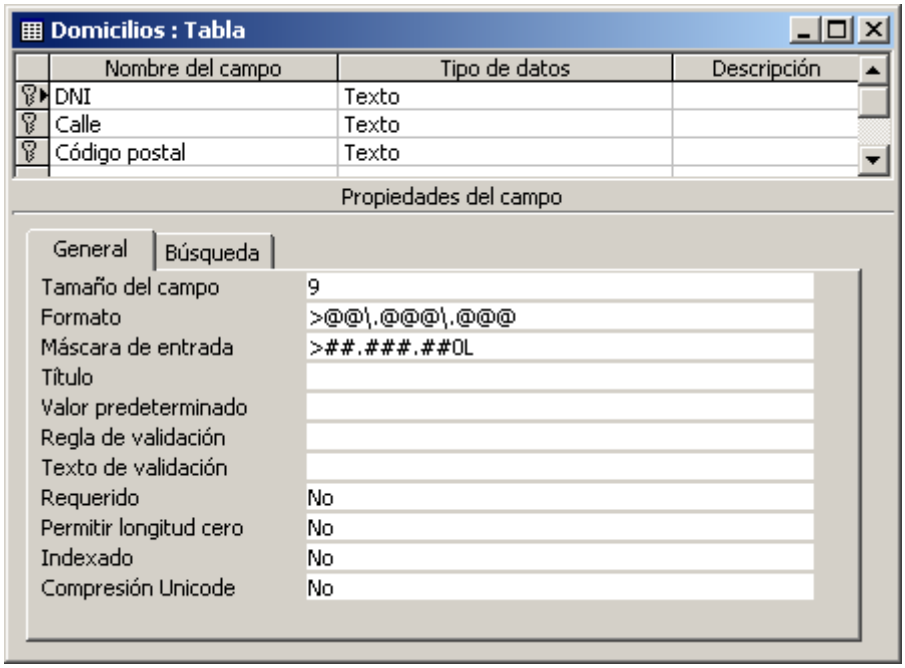
13) Agregar restricciones de integridad de dominio con las reglas de validación (el sueldo debe estar comprendido entre 0 y 120.000 €.)

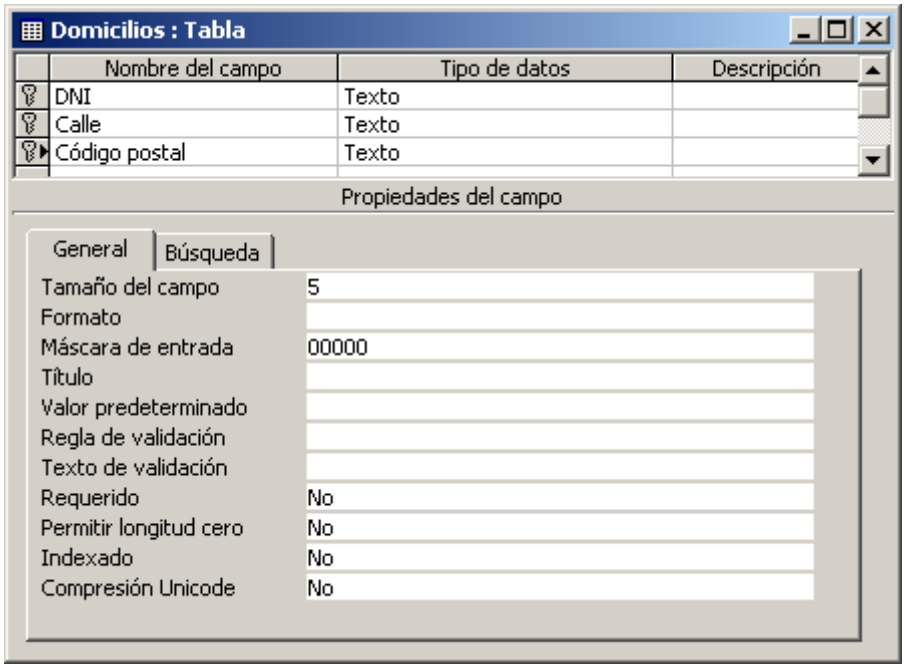


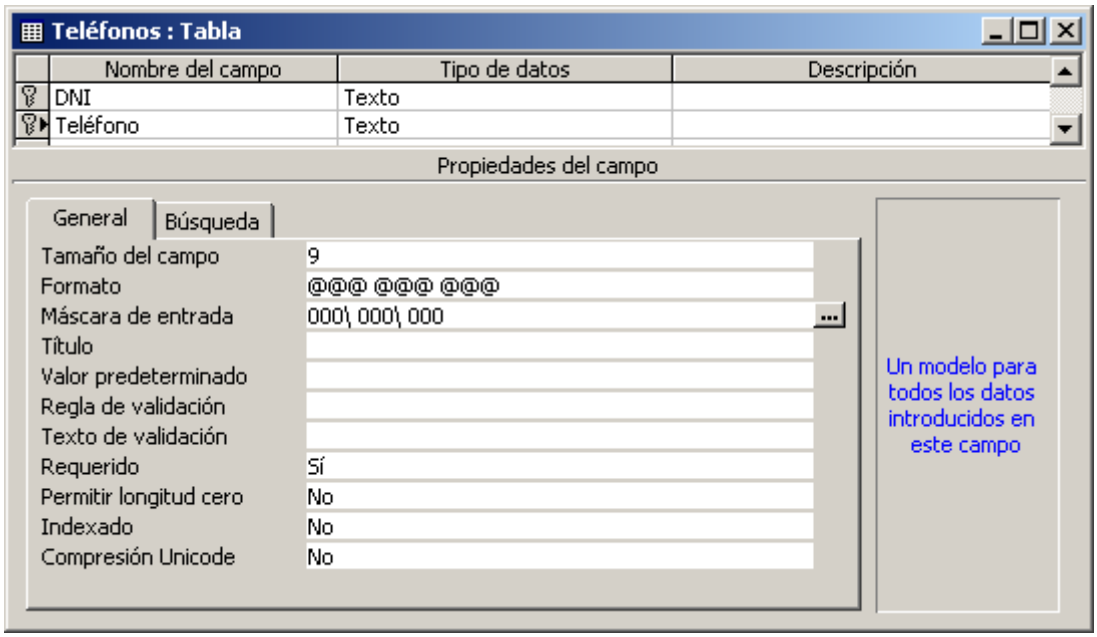
14)Agregar máscaras de entrada para todos los campos que lo acepten.

15)Agregar formato de salida para todos los campos que lo acepten.

Con respecto a estos dos apartados se pueden añadir máscaras de entrada y formato a los campos DNI, Sueldo, Código postal y Teléfono de las diferentes tablas:







El formato del campo Sueldo se puede ver en la figura del apartado 13.