Objetivo del ejercicio:

1. *Repasar el tipo de correspondencia m a n con la PK compuesta por las dos FK.*
2. *Repasar el tipo de correspondencia 1 a 1 dejando la información en dos tablas, para la primera versión.*
3. *Repasar el tipo de correspondencia 1 a 1 dejando la información en una sola tabla, para la segunda versión.*
4. *Repasar el tipo de correspondencia 1 a 0, para una tercera versión en la que consideramos que la tabla polígonos almacena todos los polígonos existentes en cada una de las ciudades almacenadas. Por lo tanto, un polígono puede tener 1 edificio de la empresa o ninguno.*
5. *Repasar modelar directamente en 3FN al detectar la entidad ciudad.*
6. *Aprender el concepto de Atributo calculado.*

   *Un **Atributo Calculado** (¡ojo!, no es derivado) es un atributo que no deberíamos almacenar, puesto que su valor puede variar frecuentemente y además, lo podemos obtener mediante una consulta. Sin embargo, en ocasiones los diseñadores deciden poner este atributo, debido a que su información se consulta con relativa frecuencia y de esta manera es más rápida la obtención de esta información. El inconveniente de almacenar el atributo calculado es que el diseñador también se tiene que encargar de que la información del mismo sea siempre correcta, esto se puede hacer fácilmente con un TRIGGER (es un procedimiento que se ejecuta automáticamente después de hacer una operación de inserción, borrado o modificación sobre una tabla).*

EJERCICIO 10. Empresa 2

Una empresa ubicada en distintos edificios de distintos polígonos industriales desea registrar la distribución de sus departamentos. Un departamento puede estar distribuido en varios edificios. Del departamento tenemos su código, nombre y el número de empleados que lo integran. De los edificios sabemos su código, nombre, dirección y el número de despachos que tienen ocupados. En cada edificio (que está localizado en un polígono industrial, y del que se conoce su código, nombre y la ciudad en la que está situado) pueden ubicarse distintos departamentos. Cada polígono industrial tiene un solo edificio de la empresa. Debido a esto, se desea controlar el número de despachos que cada departamento tiene en cada edificio.

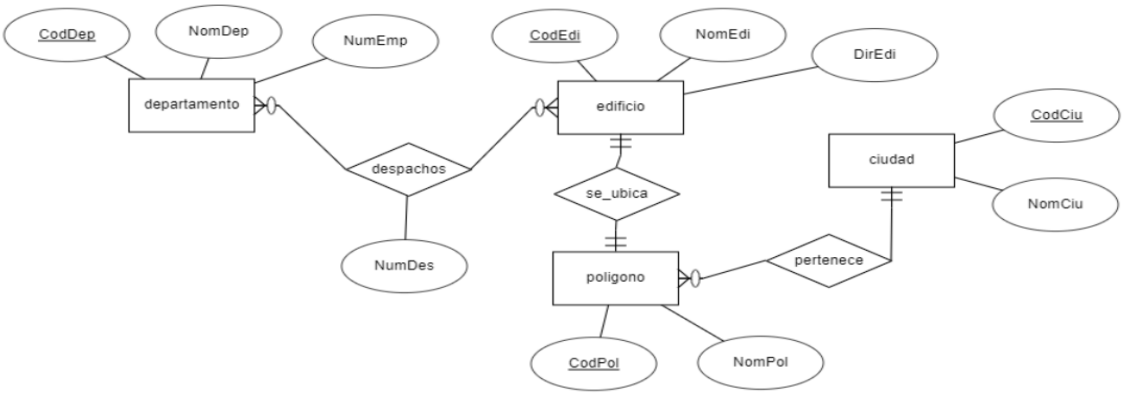
Después de leer el ejercicio podemos pensar que hay un atributo llamado, por ejemplo NumDesOcu en la tabla edificio. Este atributo podría almacenar el número de despachos ocupados que tiene cada edificio. Pero, este atributo es un **atributo calculado**, puesto que la información se puede obtener de la tabla despachos mediante una consulta. Llegados a este punto, el diseñador debe decidir si almacena el atributo porque esa información es consultada con frecuencia (en este caso, debe también encargarse de que el atributo esté siempre actualizado). O si por el contrario, no lo pone y cuando el usuario quiera saber el número de despachos que hay en un edificio, el programa ejecuta la correspondiente consulta para obtener dicha información.

En este ejercicio, para las tres versiones no lo vamos a poner y lo vamos a obtener mediante una consulta utilizando la tabla despachos.

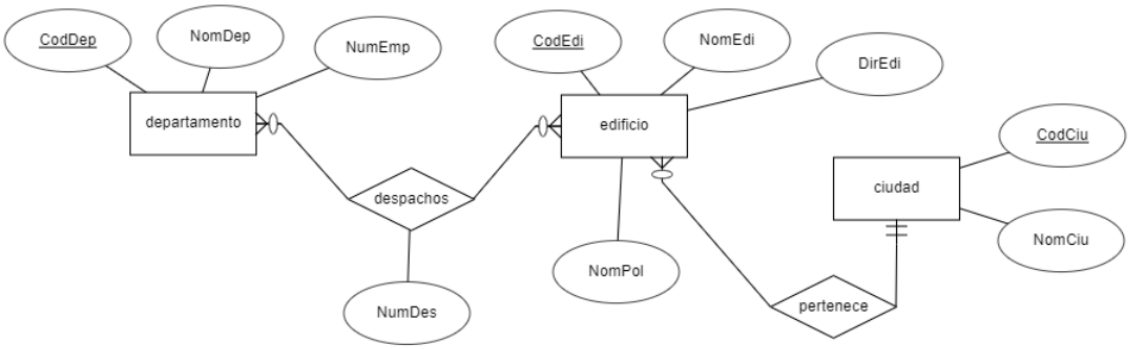
Se pide:

1. Modelar la base de datos. Para ello haremos:
   a. Diseño Conceptual de Datos utilizando un Diagrama o Modelo Entidad-Relación. Lo hacemos en papel y lo pasamos a la Herramienta CASE ERD Plus.

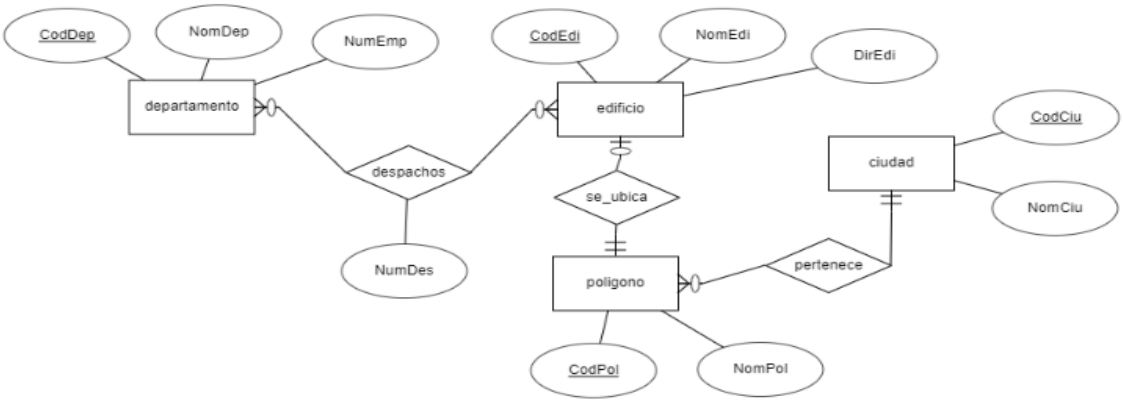
Versión 1



Versión 2



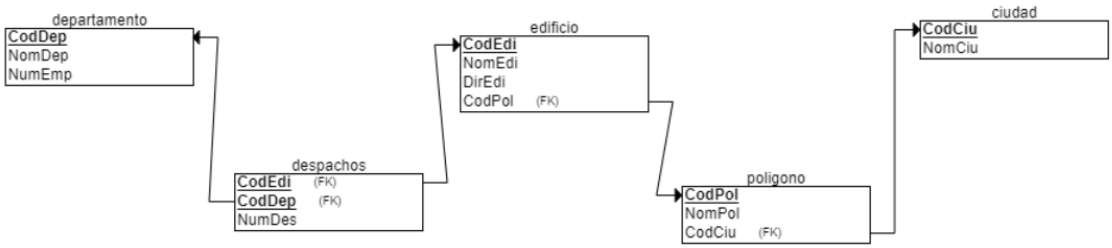
Versión 3



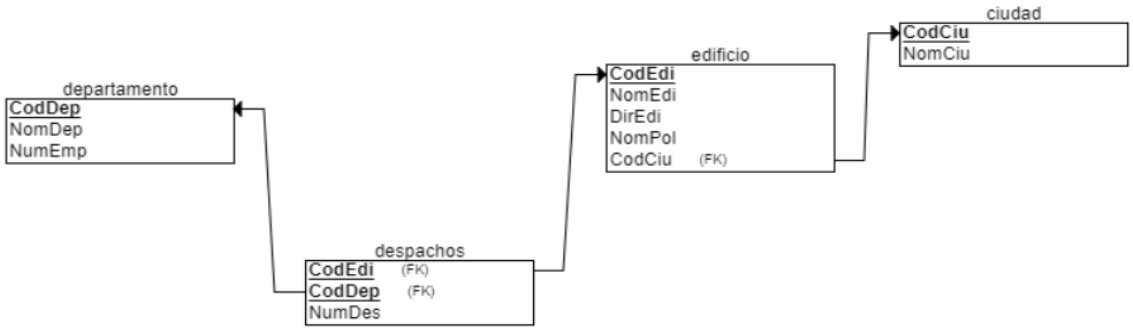
b. Diseño Lógico de Datos utilizando un Diagrama de Estructura de datos (DED). Lo hacemos en papel y lo pasamos a la Herramienta CASE MySql Workbench. En este apartado también vamos a poner el Diagrama Referencial que genera ERD Plus a partir del Modelo Entidad-Relación. Recuerda que el Diseño Lógico de Datos es hacer el modelo relacional y para ello podemos hacer un DED o un Diagrama Referencial.

Diagrama Referencial

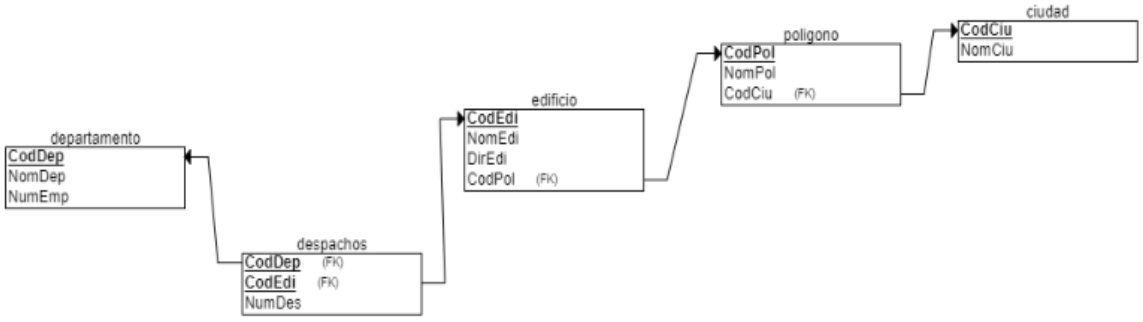
Versión 1



Versión 2

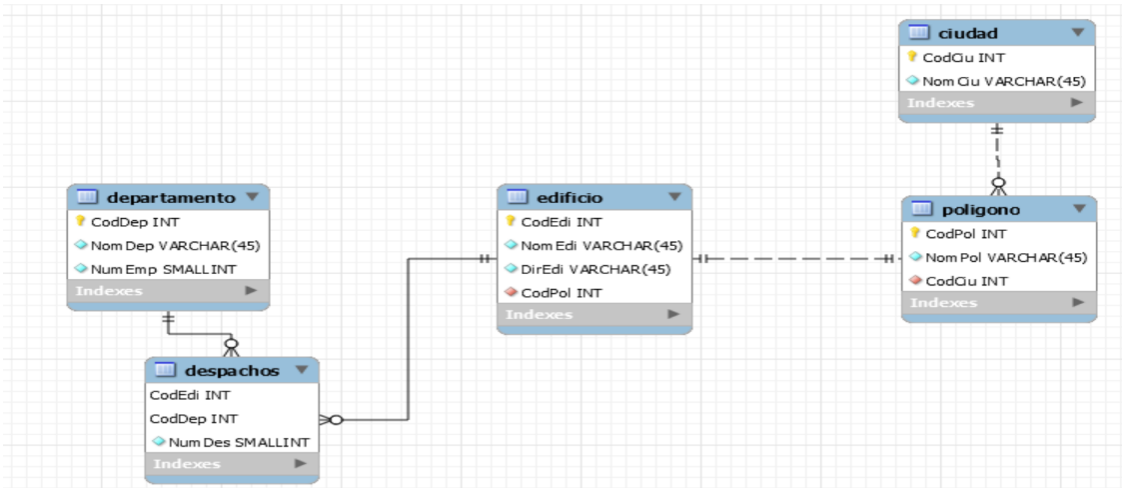


Versión 3

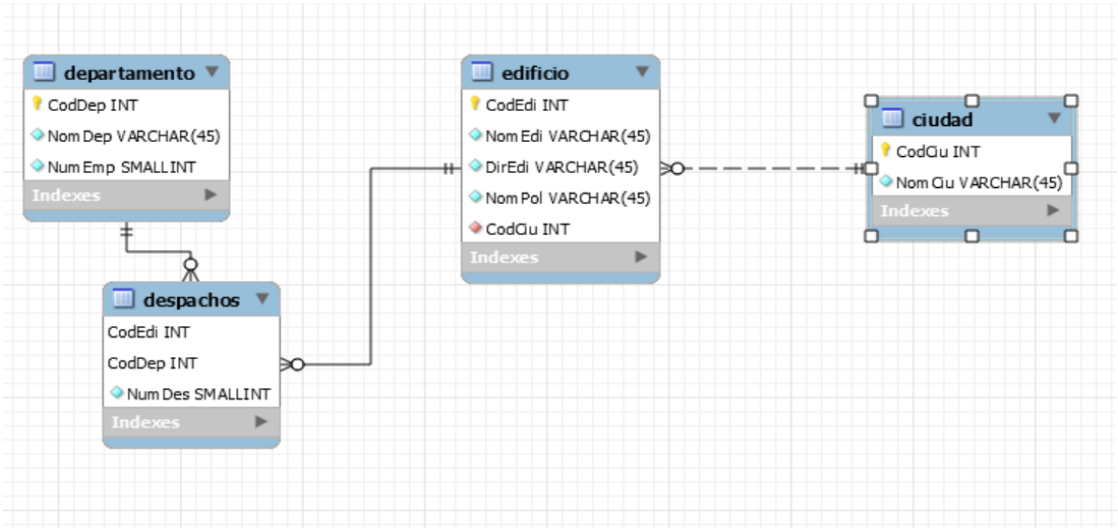


DED

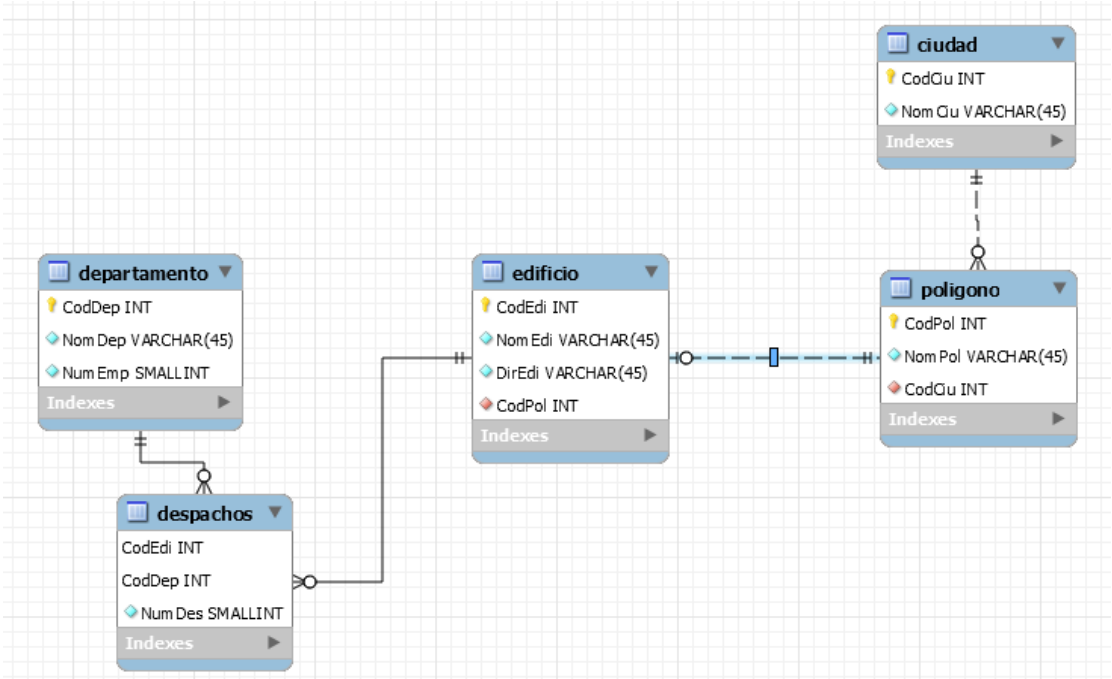
Versión 1



Versión 2



Versión 3



c. Diseño Físico de Datos. Creamos la base de datos y las tablas en SQL.

Versión 1

```
CREATE TABLE departamento
(
 CodDep INT NOT NULL,
 NomDep VARCHAR(45) NOT NULL,
 NumEmp INT NOT NULL,
 PRIMARY KEY (CodDep)
);

CREATE TABLE ciudad
(
 CodCiu INT NOT NULL,
 NomCiu VARCHAR(45) NOT NULL,
 PRIMARY KEY (CodCiu)
);

CREATE TABLE poligono
(
 CodPol INT NOT NULL,
 NomPol VARCHAR(45) NOT NULL,
 CodCiu INT NOT NULL,
 PRIMARY KEY (CodPol),
 FOREIGN KEY (CodCiu)
REFERENCES ciudad(CodCiu)
);

CREATE TABLE edificio
(
 CodEdi INT NOT NULL,
 NomEdi VARCHAR(45) NOT NULL,
 DirEdi VARCHAR(45) NOT NULL,
 CodPol INT NOT NULL,
 PRIMARY KEY (CodEdi),
 FOREIGN KEY (CodPol)
REFERENCES poligono(CodPol)
);

CREATE TABLE despachos
(
 CodEdi INT NOT NULL,
 CodDep INT NOT NULL,
 NumDes INT NOT NULL,
 PRIMARY KEY (CodDep, CodEdi),
 FOREIGN KEY (CodDep)
REFERENCES departamento(CodDep),
 FOREIGN KEY (CodEdi)
REFERENCES edificio(CodEdi)
);
```

Versión 2

```
CREATE TABLE departamento
(
 CodDep INT NOT NULL,
 NomDep VARCHAR(45) NOT NULL,
 NumEmp INT NOT NULL,
 PRIMARY KEY (CodDep)
);

CREATE TABLE ciudad
(
 CodCiu INT NOT NULL,
 NomCiu VARCHAR(45) NOT NULL,
 PRIMARY KEY (CodCiu)
);

CREATE TABLE edificio
(
 CodEdi INT NOT NULL,
 NomEdi VARCHAR(45) NOT NULL,
 DirEdi VARCHAR(45) NOT NULL,
 NomPol VARCHAR(45) NOT NULL,
 CodCiu INT NOT NULL,
 PRIMARY KEY (CodEdi),
 FOREIGN KEY (CodCiu)
REFERENCES ciudad(CodCiu)
);

CREATE TABLE despachos
(
 CodEdi INT NOT NULL,
 CodDep INT NOT NULL,
 NumDes INT NOT NULL,
 PRIMARY KEY (CodDep, CodEdi),
 FOREIGN KEY (CodDep)
REFERENCES
departamento(CodDep),
 FOREIGN KEY (CodEdi)
REFERENCES edificio(CodEdi)
);
```

Versión 3

```
CREATE TABLE departamento
(
 CodDep INT NOT NULL,
 NomDep VARCHAR(45) NOT NULL,
 NumEmp INT NOT NULL,
 PRIMARY KEY (CodDep)
);

CREATE TABLE ciudad
(
 CodCiu INT NOT NULL,
 NomCiu VARCHAR(45) NOT NULL,
 PRIMARY KEY (CodCiu)
);

CREATE TABLE poligono
(
 CodPol INT NOT NULL,
 NomPol VARCHAR(45) NOT NULL,
 CodCiu INT NOT NULL,
 PRIMARY KEY (CodPol),
 FOREIGN KEY (CodCiu)
REFERENCES ciudad(CodCiu)
);

CREATE TABLE edificio
(
 CodEdi INT NOT NULL,
 NomEdi VARCHAR(45) NOT NULL,
 DirEdi VARCHAR(45) NOT NULL,
 CodPol INT NOT NULL,
 PRIMARY KEY (CodEdi),
 FOREIGN KEY (CodPol)
REFERENCES poligono(CodPol)
);

CREATE TABLE despachos
(
 CodEdi INT NOT NULL,
 CodDep INT NOT NULL,
 NumDes INT NOT NULL,
 PRIMARY KEY (CodDep, CodEdi),
 FOREIGN KEY (CodDep)
REFERENCES
departamento(CodDep),
 FOREIGN KEY (CodEdi)
REFERENCES edificio(CodEdi)
        );
```

Observa que las tablas en las versiones 1 y 3 son exactamente iguales.
Es tarea del programador o bien del diseñador (administrador) de la base de datos, encargarse de que se cumpla:

- Para la versión 1, que un edificio está en un solo polígono y viceversa.
- Para la versión 3, que un edificio está en un solo polígono y un polígono tiene un solo edificio o ninguno. En ningún caso un polígono tiene más de un edificio.
- Para la versión 2, que el nombre del polígono en una misma ciudad no se repita.

2. Insertar datos desde phpmyadmin utilizando la sentencia INSERT INTO del LMD de SQL.

Una vez insertados datos en las tablas de ambas versiones podríamos hacer la siguiente consulta para obtener el número de despachos ocupados que tiene por ejemplo, el edificio cuyo nombre es Green Ray.

```
SELECT SUM(NumDes)
FROM edificio E JOIN despachos D ON (E.CodEdi=D.CodEdi)
WHERE NomEdi = 'Green Ray';
```