EJERCICIO 4

Objetivo del ejercicio:

1. *Repasar el tipo de correspondencia muchos a muchos (m a n).*
2. *Aprender el concepto de atributo compuesto.*
3. *Repasar el concepto de atributo derivado y opcional.*
4. *Repasar las cardinalidades mínimas*
5. *Aprender a modelar directamente en 3FN.*
6. *Aprender en SQL cómo una PK se autoincremente sola.*

Diseñar una base de datos para almacenar y gestionar información de los alumnos y las asignaturas de la ESO que refuerzan en una academia.
De los alumnos queremos conocer la siguiente información: Nombre, Apellidos (almacenar los apellidos por separado, ten en cuenta que algunos alumnos no tendrán segundo apellido), DNI, edad y dirección completa, junto con la localidad y ciudad.
De las asignaturas almacenamos el código, nombre y curso para el que se imparte. Por ejemplo, 1ºESO.
Un alumno puede reforzar muchas asignaturas en la academia. Cuando se matricula al alumno (se da de alta), se debe indicar para qué asignatura/s lo hace.
Las asignaturas que se dan en la academia pueden estar almacenadas y que aún no se hayan matriculado ningún alumno.

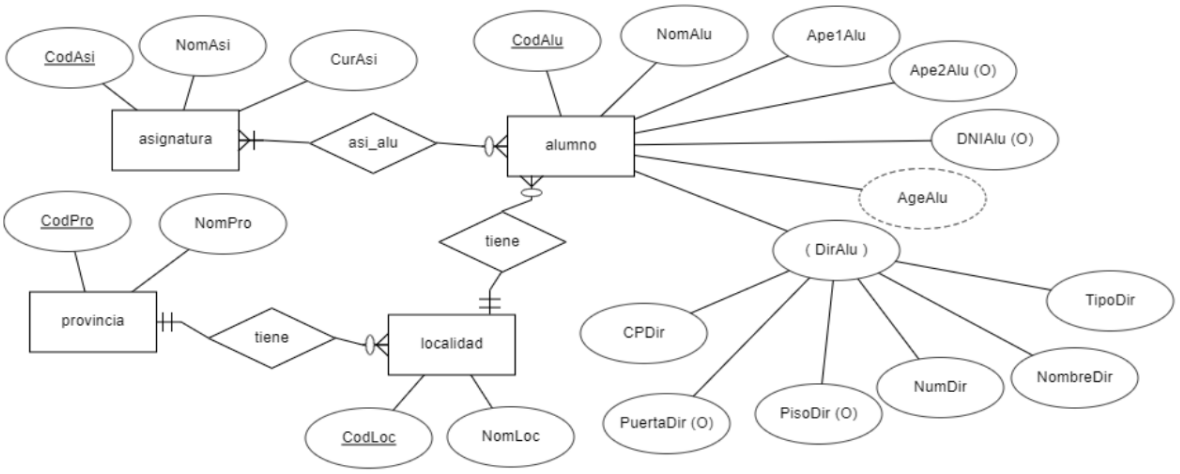
Alumno: Nombre, Apellido1, Apellido2, DNI, edad y dirección completa, junto con la localidad y ciudad.
Asignatura: código, nombre y curso.

Alumno - muchas - asignaturas (se matricula mínimo en 1)
Asignaturas - muchos - alumnos (pueden tener 0 alumnos)

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.



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.

DED:

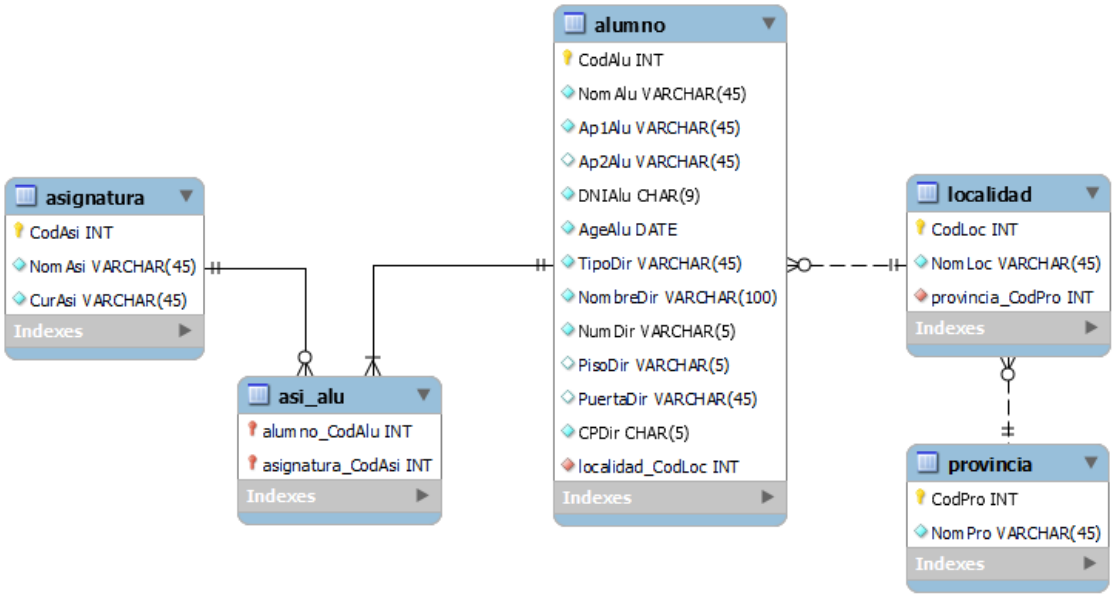
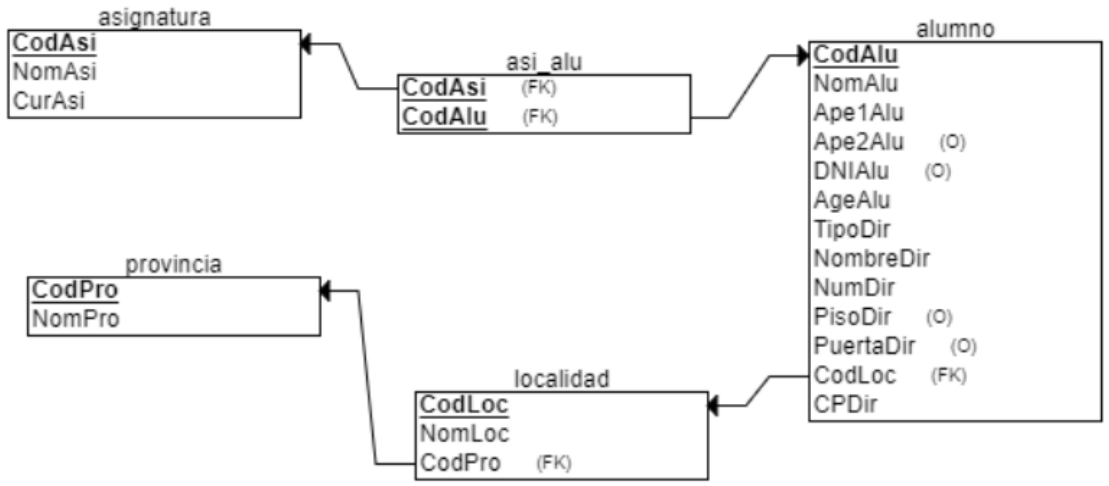


Diagrama referencial:



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

```
CREATE TABLE asignatura
(
  CodAsi INT NOT NULL,
  NomAsi VARCHAR(45) NOT NULL,
  CurAsi VARCHAR(45) NOT NULL,
  PRIMARY KEY (CodAsi)
);

CREATE TABLE provincia
(
  CodPro INT NOT NULL,
  NomPro VARCHAR(45) NOT NULL,
  PRIMARY KEY (CodPro)
);

CREATE TABLE localidad
(
  CodLoc INT NOT NULL,
  NomLoc VARCHAR(45) NOT NULL,
  CodPro INT NOT NULL,
  PRIMARY KEY (CodLoc),
  FOREIGN KEY (CodPro) REFERENCES provincia(CodPro)
);

CREATE TABLE alumno
(
  CodAlu INT NOT NULL AUTO_INCREMENT,
  NomAlu VARCHAR(45) NOT NULL,
  Ap1Alu VARCHAR(45) NOT NULL,
  Ap2Alu VARCHAR(45),
  DNIAlu CHAR(9), /* Si queremos que admita nulos y le
                  ponemos UNIQUE,solo admitirá un NULL.
                  Por lo tanto, para controlar que no se
                  repita pero pueda haber todos los NULL
                  que sean necesarios, la no repetición se
                  la podremos controlar con Trigger*/
  AgeAlu DATE NOT NULL,
  TipoDir VARCHAR(45) NOT NULL,
  NombreDir VARCHAR(100) NOT NULL,
  NumDir VARCHAR(5) NOT NULL,
  PisoDir VARCHAR(5),
  PuertaDir VARCHAR(45),
  CPDIr CHAR(5) NOT NULL,
  CodLoc INT NOT NULL,
  PRIMARY KEY (CodAlu),
  FOREIGN KEY (CodLoc) REFERENCES localidad(CodLoc)
);
```

```
CREATE TABLE asi_alu
(
  CodAlu INT NOT NULL,
  CodAsi INT NOT NULL,
  PRIMARY KEY (CodAlu, CodAsi),
  FOREIGN KEY (CodAlu) REFERENCES alumno(CodAlu),
  FOREIGN KEY (CodAsi) REFERENCES asignatura(CodAsi)
);
```

2. Insertar datos desde phpmyadmin.

```
INSERT INTO provincia (CodPro, NomPro)
VALUES (1, 'Málaga');
```

```
INSERT INTO localidad (CodLoc, NomLoc, CodPro)
VALUES (1, 'Málaga' , 1),
      (2, 'Villanueva del Trabuco' , 1),
      (3, 'Campillos' , 1),
      (4, 'Pizarra' , 1),
      (5, 'Benalmádena' , 1),
      (6, 'Torremolinos' , 1);
```

```
INSERT INTO alumno (NomAlu, Ap1Alu, Ap2Alu, DNIAlu, AgeAlu, TipoDir,
NombreDir, NumDir, PisoDir, PuertaDir, CPDir, CodLoc)
VALUES ('Santino', 'Marrero', 'Quintero', '13754975D', '2009-06-10',
'Calle', 'Barbate', '13', NULL, NULL, '29313', 2),
           ('Nacho', 'Velasco', 'Diez', '32510958K', '2009-09-25',
      'Calle', 'Compañía', '21', '5º', 'D', '29008', 1),
          ('Felix', 'Mendizábal', 'Seguí', '02915655Z', '2009-11-22',
      'Calle', 'Albéniz', '5', '1º', 'A', '29014', 1),
           ('Milagros', 'Solera', NULL, 'Y0653984Q', '2009-08-14',
      'Calle', 'Alcazabilla', '15', '7º', 'C', '29015', 1),
            ('Hernán', 'Sarmiento', 'Echeverría', '85858106A',
      '2008-07-18', 'Calle', 'Camas', '21', '3º', 'Izq', '29005', 1),
           ('Julio', 'Pastor', 'Pont', '04351793D', '2008-11-01',
      'Calle', 'de las Beatas', '3', '1º', 'A', '29008', 1),
         ('Martín', 'Puerta', 'Rodríguez', '71930788J', '2008-03-24',
      'Calle', 'Císter', '24', '3º', 'F', '29015', 1),
        ('Fernando', 'Medina', 'Recio', '12791640Y', '2008-08-05',
      'Calle', 'Córdoba', '7', '4º', 'B', '29004', 1),
           ('Sara', 'Peláez', 'Miguel', '48588954M', '2008-06-12',
      'Carrera', 'de Capuchinos', '3', '10º', 'D', '29013', 1),
          ('Juan Luis', 'Calvo', 'Galan', '72066402L', '2008-02-12',
      'Calle', 'Echegaray', '5', '5º', 'I', '29620', 6),
          ('Edmundo', 'Lloret', 'Chaves', '16100545Q', '2007-06-25',
      'Calle', 'Azucena', '15', NULL, NULL, '29320', 3),
          ('Marta', 'Sánchez', 'Ramírez', '18874991H', '2007-07-23',
      'Calle', 'Doctor López de Uralde', '4', NULL, NULL, '29560', 4),
```

```
     ('Manuel', 'Priego', 'Cortés', '55480481L', '2007-06-21',
'Calle', 'Echegaray', '10', '6º', 'D', '29015', 1),
   ('Jaime', 'Berenguer', 'Jiménez', '80938683G', '2007-07-27',
'Avenida', 'Manantial', '19', '2º', 'A', '29631', 5),
     ('Germán', 'de Gutiérrez', '','96022708P', '2006-08-22',
'Calle', 'de la Cruz', '2', '7º', 'B', '29620', 6),
    ('Elena', 'Flores', 'Corominas', '12345678A', '2006-05-21',
'Calle', 'los Gavilanes', '44', '5º', 'A', '29140', 1),
     ('Nicolás', 'Méndez', 'Rincón', '09814692V', '2006-07-12',
'Calle', 'Pilar Lorengar', '7', '9º', 'C', '29004', 1),
      ('Juan', 'Ibáñez', 'Marín', '88039011W', '2005-07-18',
'Calle', 'Casarabonela', '12', '2º', 'C', '29006', 1),
   ('Esmeralda', 'Clemente', 'Bayo', '62817670R', '2006-12-18',
'Calle', 'Joaquín de Molina', '6', NULL, NULL, '29016', 1),
        ('José Manuel', 'Belda', 'Vázquez', '41632110V',
'2005-02-18', 'Calle', 'Félix Revello de Toro', '79', NULL,
NULL, '29016', 1);
```

```
INSERT INTO asignatura (CodAsi, NomAsi, CurAsi)
VALUES (1, 'Lengua y literatura', '1ºESO'),
      (2, 'Lengua y literatura', '2ºESO'),
      (3, 'Lengua y literatura', '3ºESO'),
      (4, 'Lengua y literatura', '4ºESO'),
      (5, 'Matemáticas', '1ºESO'),
      (6, 'Matemáticas', '2ºESO'),
      (7, 'Matemáticas', '3ºESO'),
      (8, 'Matemáticas', '4ºESO'),
      (9, 'Física y química', '1ºESO'),
      (10, 'Física y química', '2ºESO'),
      (11, 'Física y química', '4ºESO'),
      (12, 'Biología', '1ºESO'),
      (13, 'Biología', '3ºESO'),
      (14, 'Biología', '4ºESO'),
      (15, 'Historia', '4ºESO');
```

```
INSERT INTO asi_alu (CodAlu, CodAsi)
VALUES (1,1), (1,5), (1,9), (1,12), (2,9),
        (3,5), (3,9), (3,12), (4,1), (4,12),
        (5,5), (5,9), (5,12),
        (6,2), (6,10), (7,2), (7,6), (7,10),
        (8,6), (8,10), (9,10), (10,6), (10,10),
        (11,7), (11,13), (12,3), (12,7), (12,13),
        (13,7), (14,3), (14,7), (14,13), (15,3),
        (16,8), (16,11), (17,4), (17,8), (17,11),
        (17,14), (18,11), (19,8), (19,14), (20,4),
        (20,11), (20,14);
```

3. Realizar las siguientes consultas en SQL:

- Muestra todas las filas y todos los campos de las tablas. Ordenar el resultado de la consulta.

```
SELECT *
FROM provincia;

SELECT *
FROM localidad;

SELECT *
FROM alumno
ORDER BY Ap1Alu;


SELECT *
FROM asignatura
ORDER BY CurAsi;

SELECT *
FROM asi_alu
ORDER BY CodAsi;
```

- Muestra algunos campos de las tablas.

```
SELECT NomAlu, Ap1Alu, Ap2Alu, YEAR(SYSDATE())-YEAR(AgeAlu)
FROM alumno;

SELECT NomAsi, CurAsi
FROM asignatura;

SELECT CodAsi
FROM asi_alu
ORDER BY CodAsi;
```

- Mostrar para cada asignatura los alumnos que están matriculados. Muestra primero toda la información y posteriormente muestra solo el nombre de la asignatura y nombre con apellidos del alumno. ¿Qué ocurre con las asignaturas que aún no tienen alumnos matriculados?

```
SELECT * FROM asignatura;
SELECT * FROM asi_alu ORDER BY CodAsi;


SELECT *
FROM asignatura asi INNER JOIN asi_alu asal ON
(asi.CodAsi=asal.CodAsi);

SELECT *
FROM (asignatura asi INNER JOIN asi_alu asal ON
(asi.CodAsi=asal.CodAsi))
INNER JOIN alumno alu ON (alu.CodAlu=asal.CodAlu);
```

EJERCICIO 4

```
SELECT NomAsi, NomAlu, Ap1Alu, Ap2Alu
FROM (asignatura asi INNER JOIN asi_alu asal ON
(asi.CodAsi=asal.CodAsi))
INNER JOIN alumno alu ON (alu.CodAlu=asal.CodAlu);
```

EJERCICIO 4

Las asignaturas que no tienen alumnos matriculados no aparecen. Para que aparezcan, tenemos que utilizar `RIGHT JOIN` o `LEFT JOIN` adecuadamente:

```
/*Queremos que salga los valores de la tabla de la izquierda
aunque no tengan valor de combinación*/

SELECT *
FROM asignatura asi LEFT OUTER JOIN asi_alu asal ON
(asi.CodAsi=asal.CodAsi);


/*Queremos que salga los valores de la tabla de la izquierda
aunque no tengan valor de combinación. Ahora la tabla de la
izquierda es:(asignatura asi LEFT OUTER JOIN asi_alu asal ON
(asi.CodAsi=asal.CodAsi))*/

SELECT *
FROM (asignatura asi LEFT OUTER JOIN asi_alu asal ON
(asi.CodAsi=asal.CodAsi))
LEFT OUTER JOIN alumno alu ON (alu.CodAlu=asal.CodAlu)
ORDER BY asi.CodAsi;

/*Finalmente se nos queda así:*/

SELECT NomAsi, NomAlu, Ap1Alu, Ap2Alu
FROM (asignatura asi LEFT OUTER JOIN asi_alu asal ON
(asi.CodAsi=asal.CodAsi))
LEFT OUTER JOIN alumno alu ON (alu.CodAlu=asal.CodAlu)
ORDER BY asi.CodAsi;
```

EJERCICIO 4

- Mostrar para cada alumno cuáles son las asignaturas que refuerza. Muestra primero toda la información y posteriormente muestra solo el nombre y apellidos del alumno junto con el nombre de la asignatura. Ordenar ascendentemente por asignatura y dentro de cada asignatura, también ascendentemente por apellidos y nombre del alumno.

```
SELECT *
FROM (alumno alu INNER JOIN asi_alu asal ON
(alu.CodAlu=asal.CodAlu))
INNER JOIN asignatura asi ON (asi.CodAsi=asal.CodAsi);


SELECT NomAlu, Ap1Alu, Ap2Alu, NomAsi
FROM (alumno alu INNER JOIN asi_alu asal ON
(alu.CodAlu=asal.CodAlu))
INNER JOIN asignatura asi ON (asi.CodAsi=asal.CodAsi)
ORDER BY NomAsi, Ap1Alu, Ap2Alu, NomAlu ASC;
```

- Para la asignatura de Matemáticas de 1º de la ESO, mostrar el nombre y apellidos de los alumnos que están matriculados.

```
SELECT *
FROM (alumno alu INNER JOIN asi_alu asal ON
(alu.CodAlu=asal.CodAlu))
INNER JOIN asignatura asi ON (asi.CodAsi=asal.CodAsi)
WHERE NomAsi = 'Matemáticas' AND CurAsi = '1ºESO';

SELECT NomAlu, Ap1Alu, Ap2Alu
FROM (alumno alu INNER JOIN asi_alu asal ON
(alu.CodAlu=asal.CodAlu))
INNER JOIN asignatura asi ON (asi.CodAsi=asal.CodAsi)
WHERE NomAsi = 'Matemáticas' AND CurAsi = '1ºESO';
```

- Para el alumno con DNI 12345678A, mostrar las asignaturas que refuerza.

```
SELECT NomAsi
FROM (alumno alu INNER JOIN asi_alu asal ON
(alu.CodAlu=asal.CodAlu))
INNER JOIN asignatura asi ON (asi.CodAsi=asal.CodAsi)
WHERE DNIAlu = '12345678A';
```