

# Administración de base de datos

Proyecto final

Tecnicatura Superior en programación

UTN – Extensión Áulica Chivilcoy

# 2022

Agustin Narbebury

agusn11@hotmail.com

---

Noviembre 2022

# **Índice.**

<i>Introducción</i>	<i>1</i>
<i>Contexto</i>	<i>2</i>
<i>Recopilación y análisis de requisitos</i>	<i>2</i>
<i>Documento de detalle</i>	<i>7</i>
<i>Descripción de los procesos</i>	<i>10</i>
<i>Roles de usuarios</i>	<i>17</i>
<i>Análisis de la DMS</i>	<i>17</i>
<i>Modelo E-R</i>	<i>18</i>
<i>Modelo Relacional y Dependencias</i>	<i>19</i>
<i>Algebra Relacional</i>	<i>20</i>
<i>Definición de tablas SQL</i>	<i>22</i>
<i>Carga de Datos</i>	<i>26</i>
<i>Consultas SQL</i>	<i>30</i>
<i>Usuarios</i>	<i>32</i>

## **Introducción.**

*El objetivo de este trabajo es aplicar en la metodología estudiada en la materia con la finalidad de que sirva como punto de partida y modelo para diseñar bases de datos y poder decidir cuáles son las mejores alternativas para implementarlas, incorporando paulatinamente los conceptos teóricos y entendiendo su aplicación práctica.*

*Se plantea un caso de trabajo para llevar a cabo un análisis, administración y gestión de un sistema de alumnos para una institución educativa.*

*Este proyecto aborda la realización de un sistema de base de datos para administrar alumnos con sus datos y estado académicos en una escuela de manera eficaz y fiable, dado que muchas instituciones educativas no cuentan con un buen manejo de datos.*

*Es necesario dejar en claro que el contexto descripto, las afirmaciones hechas en este y los datos utilizados en este trabajo no reflejan la realidad con exactitud y han sido definidos solo con el objetivo de verificar el correcto proceso de análisis, diseño y funcionamiento de la base de datos implementada.*

## **Contexto.**

*Este proyecto tiene como finalidad la implementación de un sistema para gestión y administración de Alumnos en una institución educativa.*

*El sistema llevara un seguimiento cuidadoso de toda la información que se implementara sobre los alumnos, profesores y carreras que se encuentran disponibles.*

## **Recopilación y análisis de requisitos.**

*Con el propósito de recopilar los requerimientos del software para la consolidación de información y una correcta gestión de los alumnos, enfocado en la base de datos concluyo los siguientes requerimientos funciones para el sistema:*

- Registrar al usuario administrador del sistema.*
- Cargar carreras.*
- Cargar materias.*
- Cargar aulas*
- Dar de alta a un profesor en el sistema.*
- Dar de alta a un alumno en el sistema.*
- Cargar provincia del alumno.*
- Cargar ciudad del alumno.*
- Cargar calle del alumno.*
- Cargar provincia del profesor.*
- Cargar ciudad del profesor.*
- Cargar calle del profesor.*
- Cargar notas de los alumnos.*

- Actualizar notas de los alumnos.
- Actualizar materia de los alumnos.
- Actualizar carrera de los alumnos.
- Actualizar materia de los profesores.
- Actualizar carrera de los profesores.
- Actualizar aula.
- Actualizar provincia del profesor.
- Actualizar ciudad del profesor.
- Actualizar calle del profesor.
- Actualizar provincia del alumno.
- Actualizar ciudad del alumno.
- Actualizar calle del alumno.
- Eliminar carreras.
- Eliminar alumnos.
- Eliminar profesores.
- Eliminar aulas.
- Realizar búsqueda de alumno.
- Realizar búsqueda de profesores.
- Realizar búsqueda de carreras.
- Consultar el estado académico del alumno.

*La gestión de información hace referencia a los alumnos, con sus profesores, materias, carrera y aula.*

*Para poder realizar un seguimiento y lograr un correcto uso del sistema es necesario que los administradores del sistema cuenten ciertos datos.*

**Se solicitará al usuario administrador del sistema los siguientes datos:**

**Carrera.**

- ✓ **COD\_Carrera**
- ✓ **Nombre**
- ✓ **Duración**

**Materia.**

- ✓ **COD\_Materia**
- ✓ **Nombre**
- ✓ **COD\_Carrera1**

**Aula.**

- ✓ **COD\_Aula**
- ✓ **Capacidad**

**Persona.**

- ✓ **Nombre**
- ✓ **Apellido**
- ✓ **Fecha\_Nacimiento**
- ✓ **DNI**
- ✓ **Telefono**
- ✓ **COD\_Provincia2**
- ✓ **COD\_Ciudad2**
- ✓ **COD\_Calle1**

### **Provincia.**

- ✓ COD\_Provincia
- ✓ Provincia

### **Ciudad.**

- ✓ COD\_Ciudad
- ✓ Ciudad
- ✓ COD\_Provincia1

### **Calle.**

- ✓ COD\_Calle
- ✓ Calle
- ✓ Numero
- ✓ COD\_Ciudad1

### **Alumno**

- ✓ *Matricula\_Alumno*
- ✓ *DNI1*
- ✓ *COD\_Carrera2*

### **Profesor**

- ✓ *COD\_Profesor*
- ✓ *Horario*
- ✓ *DNI2*

**Nota.**

- ✓ **Nota**
- ✓ **Faltas**
- ✓ **Matricula\_Alumno2**
- ✓ **COD\_Materia1**

**Estudia.**

- ✓ **COD\_Profesor1**
- ✓ **COD\_Materia2**
- ✓ **Matricula\_Alumno3**
- ✓ **COD\_Aula1**

**Ejerce.**

- ✓ **COD\_Profesor2**
- ✓ **COD\_Carrera3**



## **Documento de detalles.**

### **Carrera**

- **COD\_Carrera:** Número entero TINYINT (Max.127) **PRIMARY KEY**
- **Nombre:** Texto máximo 60 caracteres NOT NULL.
- **Duración (años):** Float

### **Materia**

- **COD\_Materia:** Número entero (INT) **PRIMARY KEY**
- **Nombre:** Texto máximo 50 caracteres NOT NULL
- **COD\_Carrera1:** **FOREIGN KEY (TINYINT)** NOT NULL

### **Aula.**

- **COD\_Aula:** Número entero TINYINT (Max.127) **PRIMARY KEY**
- **Capacidad:** Número entero TINYINT (Max.127) NOT NULL

### **Persona.**

- **Nombre:** Texto máximo 60 caracteres. NOT NULL
- **Apellido:** Texto máximo 60 caracteres. NOT NULL
- **Fecha\_Nacimiento:** Dato de tipo fecha (DATE)
- **DNI:** BIGINT **PRIMARY KEY**
- **Telefono:** BIGINT NOT NULL
- **COD\_Provincia2:** **FOREIGN KEY (TINYINT)** NOT NULL
- **COD\_Ciudad2:** **FOREIGN KEY (INT)** NOT NULL
- **COD\_Calle1:** **FOREIGN KEY (TINYINT)** NOT NULL

### Provincia.

- **COD\_Provincia:** Número entero TINYINT (Max.127) **PRIMARY KEY**
- **Provincia:** Texto máximo 51 caracteres. NOT NULL

### Ciudad.

- **COD\_Ciudad:** Número entero INT **PRIMARY KEY**
- **Ciudad:** Texto máximo 60 caracteres NOT NULL
- **COD\_Provincia1:** **FOREIGN KEY (TINYINT)** NOT NULL

### Calle.

- **COD\_Calle:** Número entero INT **PRIMARY KEY**
- **Calle:** Texto máximo 40 caracteres NOT NULL
- **Numero:** Número entero SMALLINT NOT NULL
- **COD\_Ciudad1:** **FOREIGN KEY (INT)** NOT NULL

### Alumno

- **Matricula\_Alumno:** Número entero (INT).
- **DNI1:** **FOREIGN KEY (BIGINT)** NOT NULL
- **COD\_Carrera2:** **FOREIGN KEY (TINYINT)** PRIMARY KEY NOT NULL

### Profesor

- **COD\_Profesor:** Número entero (INT) **PRIMARY KEY**
- **Horario:** Texto máximo 10 caracteres NOT NULL
- **DNI2:** **FOREIGN KEY (BIGINT)** NOT NULL

### **Nota.**

- **Nota:** *FLOAT*
- **Faltas:** *Número entero TINYINT (MAX 127) NOT NULL*
- **Matricula\_Alumno2:** **FOREIGN KEY** *Número entero INT NOT NULL*
- **COD\_Materia1:** **FOREIGN KEY** *Número entero (INT) NOT NULL*

### **Estudia.**

- **COD\_Profesor1:** **FOREIGN KEY** *Número entero (INT) NOT NULL*
- **COD\_Materia2:** **FOREIGN KEY** *Número entero (INT) NOT NULL* **Matricula\_Alumno3:** **FOREIGN KEY** *Número entero INT NOT NULL*
- **COD\_Aula1:** **FOREIGN KEY** *Número entero TINYINT NOT NULL*

### **Ejerce.**

- **COD\_Profesor2:** **FOREIGN KEY** *Número entero (INT) NOT NULL*
- **COD\_Carrera3:** **FOREIGN KEY** *Número entero (TINYINT) NOT NULL*

## **Descripción de los procesos.**

### **Carga de carrera.**

Su función consiste en registrar una nueva carrera en la base de datos. Para ello el usuario debe ingresar los datos en un formulario el cual esta sincronizado con la BD.

### **Entrada.**

- ✓ *COD\_Carrera*
- ✓ *Nombre*
- ✓ *Duración*

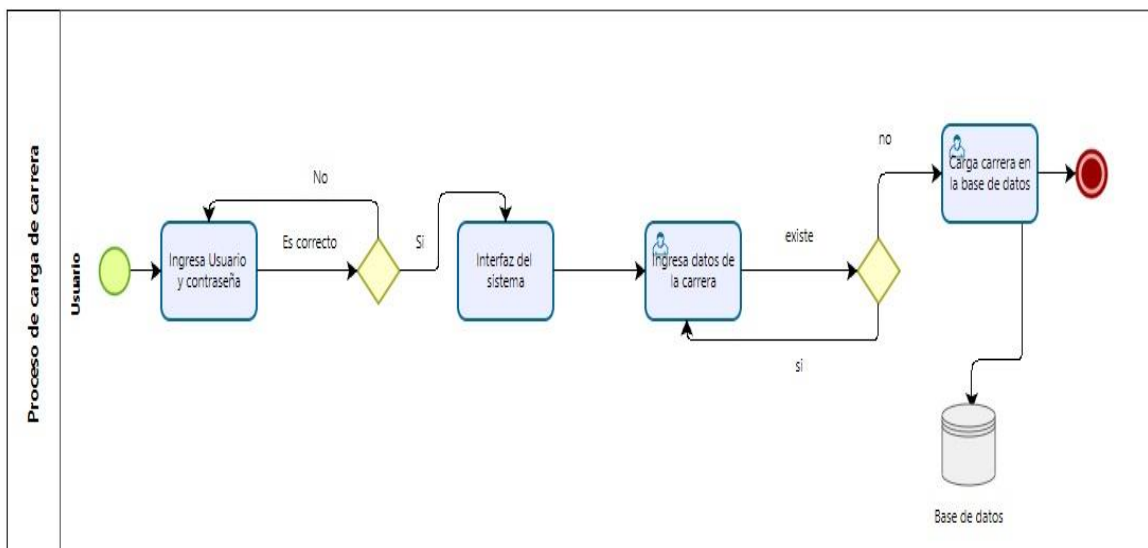
### **Proceso.**

Una vez haya accedido a esta utilidad del sistema se introducen todos los datos de la carrera, estos serán validados por el sistema, comprobando la existencia previa de la carrera en la BD, o en su efecto generando un nuevo registro.

Cualquier error que se produzca durante el proceso debe ser notificado al usuario mediante un mensaje descriptivo en pantalla.

### **Salida.**

Registro de la BD.



## Carga de materia.

Su función consiste en registrar una nueva materia en la base de datos. Para ello el usuario debe ingresar los datos en un formulario el cual esta sincronizado con la BD.

## Entrada.

- ✓ *COD\_Materia*
- ✓ *Nombre*
- ✓ *COD\_Carrera1*

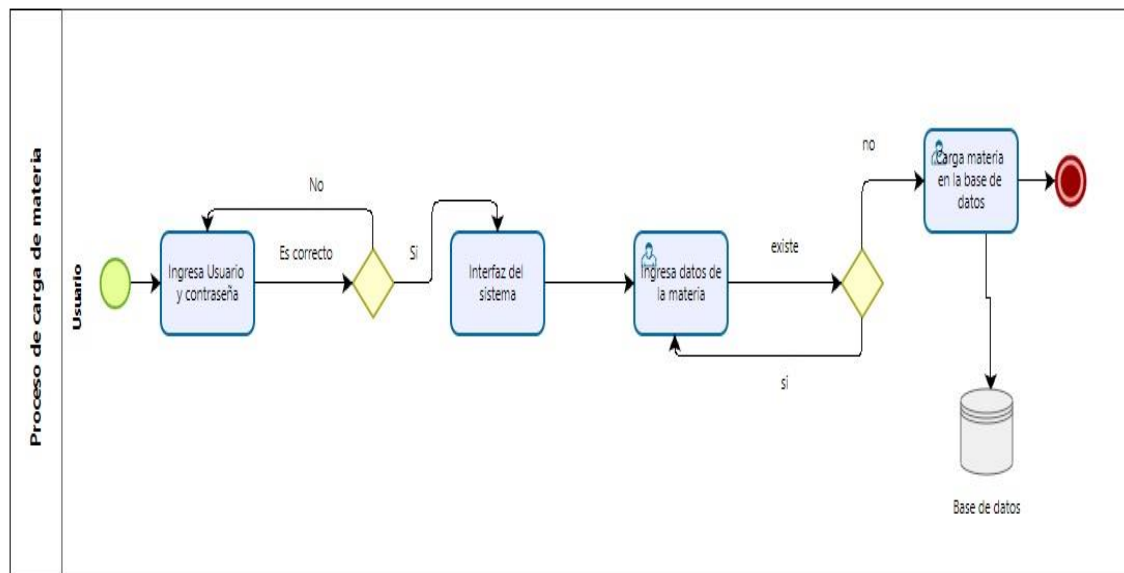
## Proceso.

Una vez haya accedido a esta utilidad del sistema se introducen todos los datos de la materia, estos serán validados por el sistema, comprobando la existencia previa de la materia en la BD, o en su efecto generando un nuevo registro.

Cualquier error que se produzca durante el proceso debe ser notificado al usuario mediante un mensaje descriptivo en pantalla.

## Salida.

Registro de la BD.



## Carga de aula.

Su función consiste en registrar una nueva aula en la base de datos. Para ello el usuario debe ingresar los datos en un formulario el cual esta sincronizado con la BD.

## Entrada.

- ✓ *COD\_Aula*
- ✓ *Capacidad*

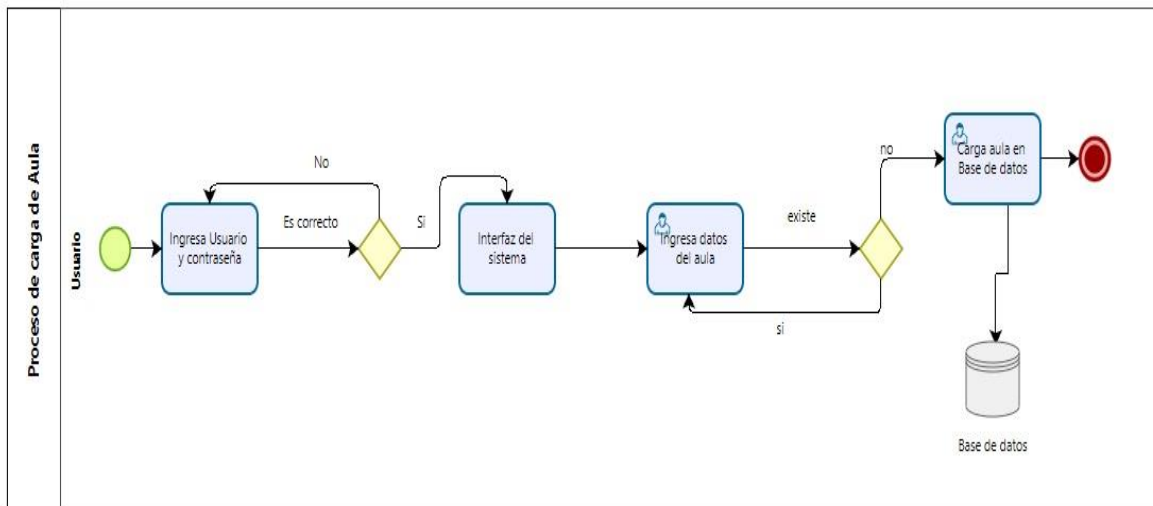
## Proceso.

Una vez haya accedido a esta utilidad del sistema se introducen todos los datos del aula, estos serán validados por el sistema, comprobando la existencia previa del aula en la BD, o en su efecto generando un nuevo registro.

Cualquier error que se produzca durante el proceso debe ser notificado al usuario mediante un mensaje descriptivo en pantalla.

## Salida.

Registro de la BD.



## Carga de persona.

Su función consiste en registrar una nueva persona en la base de datos. Para ello el usuario debe ingresar los datos en un formulario el cual esta sincronizado con la BD.

## Entrada.

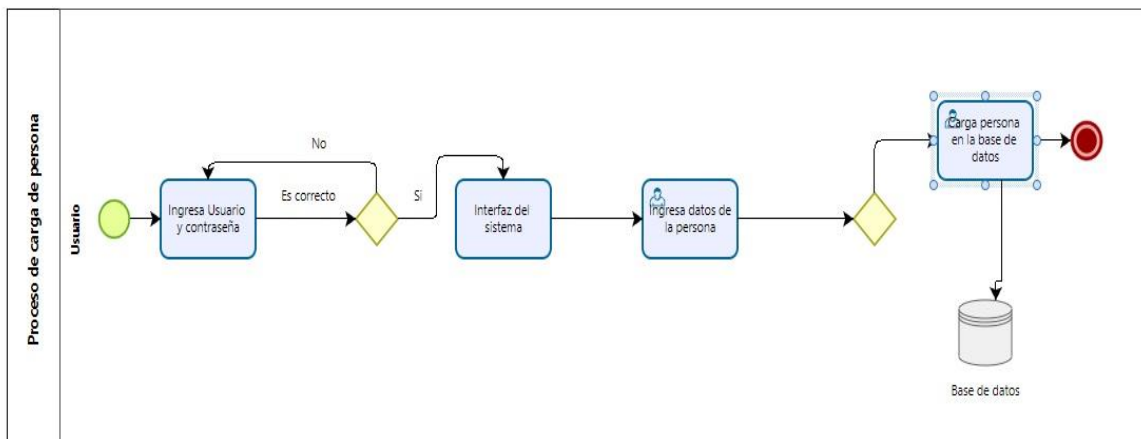
- ✓ **Nombre**
- ✓ **Apellido**
- ✓ **Fecha\_Nacimiento**
- ✓ **DNI**
- ✓ **Telefono**
- ✓ **COD\_Provincia2**
- ✓ **COD\_Ciudad2**
- ✓ **COD\_Calle1**

## Proceso.

Una vez haya accedido a esta utilidad del sistema se introducen todos los datos de la persona, estos serán validados por el sistema, comprobando la existencia previa de la persona en la BD, o en su efecto generando un nuevo registro. Cualquier error que se produzca durante el proceso debe ser notificado al usuario mediante un mensaje descriptivo en pantalla.

## Salida.

Registro de la BD.



## Carga de alumno.

Su función consiste en registrar un nuevo alumno en la base de datos. Para ello el usuario debe ingresar los datos en un formulario el cual esta sincronizado con la BD.

## Entrada.

- ✓ Matricula\_Alumno
- ✓ DNI1
- ✓ COD\_Carrera1

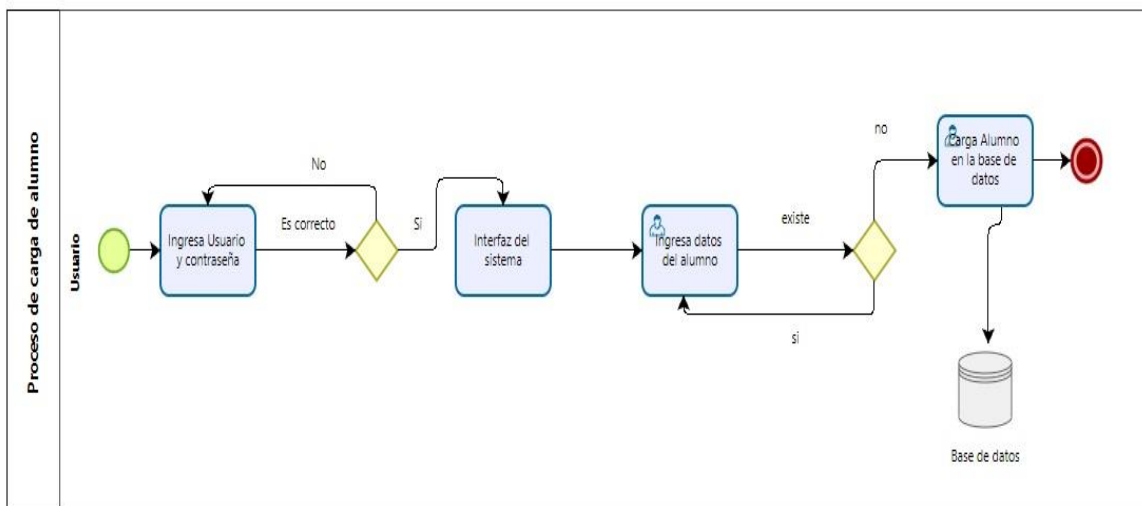
## Proceso.

Una vez haya accedido a esta utilidad del sistema se introducen todos los datos del alumno, estos serán validados por el sistema, comprobando la existencia previa del alumno en la BD, o en su efecto generando un nuevo registro.

Cualquier error que se produzca durante el proceso debe ser notificado al usuario mediante un mensaje descriptivo en pantalla.

## Salida.

Registro de la BD.





## Carga de profesor

Su función consiste en registrar un nuevo profesor en la base de datos. Para ello el usuario debe ingresar los datos en un formulario el cual esta sincronizado con la BD.

### Entrada

- ✓ COD\_Profesor
- ✓ DNI2
- ✓ *Horario*

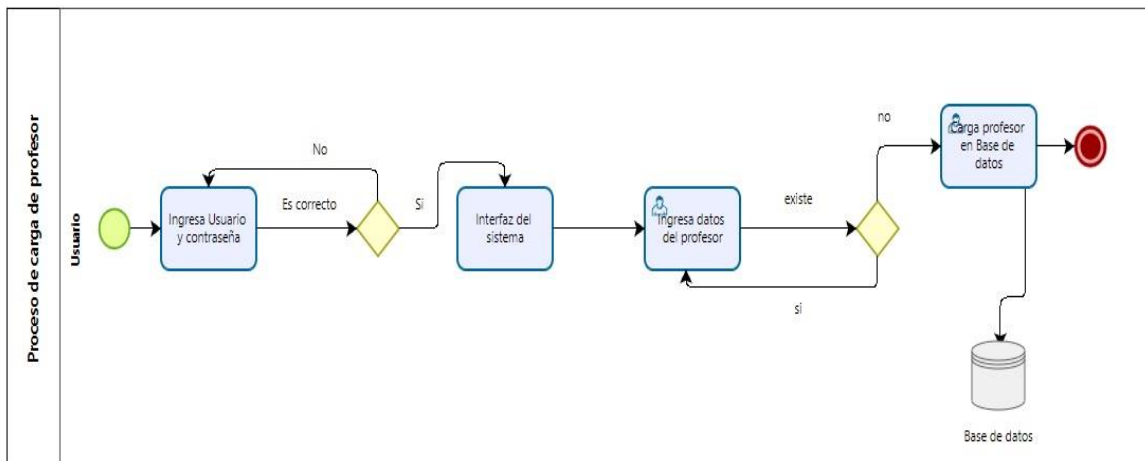
### Proceso.

Una vez haya accedido a esta utilidad del sistema se introducen todos los datos del profesor, estos serán validados por el sistema, comprobando la existencia previa del alumno en la BD, o en su efecto generando un nuevo registro.

Cualquier error que se produzca durante el proceso debe ser notificado al usuario mediante un mensaje descriptivo en pantalla.

### Salida

Registro de la BD.



## Carga de notas

Su función consiste en registrar una nueva nota en la base de datos. Para ello el usuario debe ingresar los datos en un formulario el cual esta sincronizado con la BD.

### Entrada

- ✓ Nota
- ✓ Faltas
- ✓ *Matricula\_Alumno2*
- ✓ *COD\_Materia1*

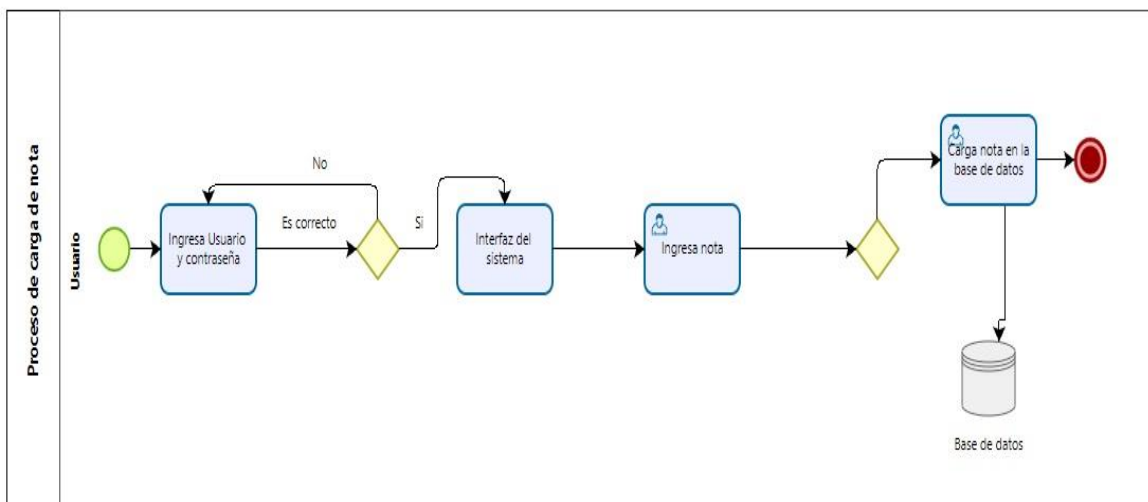
### Proceso.

Una vez haya accedido a esta utilidad del sistema se introducirá la nota.

Cualquier error que se produzca durante el proceso debe ser notificado al usuario mediante un mensaje descriptivo en pantalla.

### Salida

Registro de la BD.



## **Roles de los usuarios.**

*Solamente podrán tener acceso al sistema y disponer de los permisos para modificar los datos los preceptores que serán usuarios administradores encargados de manejar todos los datos.*

## **Análisis de la DBMS Elegida: MariaDB.**

*MariaDB es un sistema de base de datos que proviene de MySQL, desarrollado por Michael Widenius, fundador de MySQL y la comunidad de software libre.*

*Su licencia es GPL (General Public License) la cual ofrece la capacidad de usarlo, compartirlo y modificarlo.*

*MariaDB tiene las mismas funciones que MySQL a través de Access control lists para acceder a todos los objetos y operaciones.*

*Ofrece mejoras en los motores de almacenamiento.*

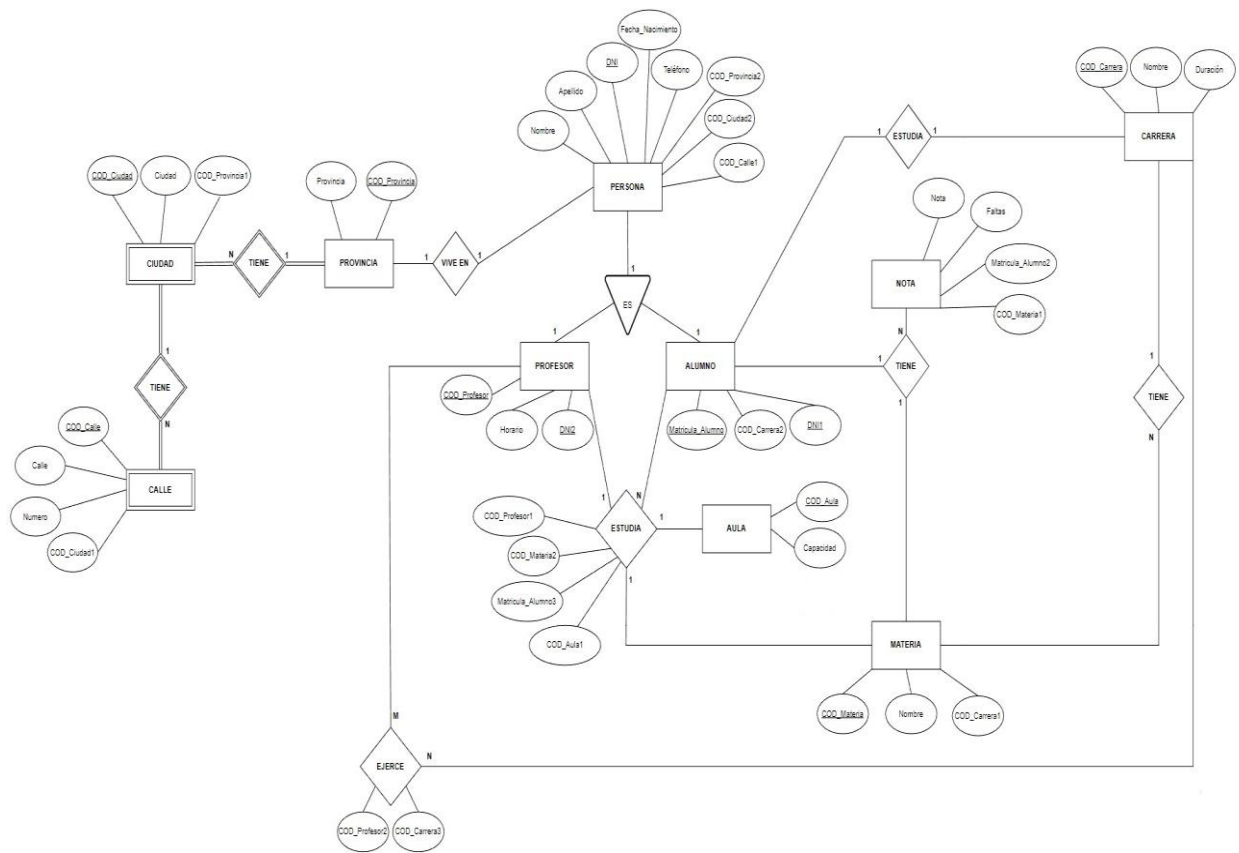
## **Las ventajas de esta DBMS son:**

- *Diseño para que el usuario de MySQL pueda migrar sin complicaciones, un cliente de MariaDB puede conectarse a un servidor MySQL y viceversa.*
- *Mejor estabilidad y velocidad.*
- *Uso de “columnas virtuales”.*
- *Es libre, licencia GNP-GPL.*
- *Implementa Stored Procedures y Triggers.*

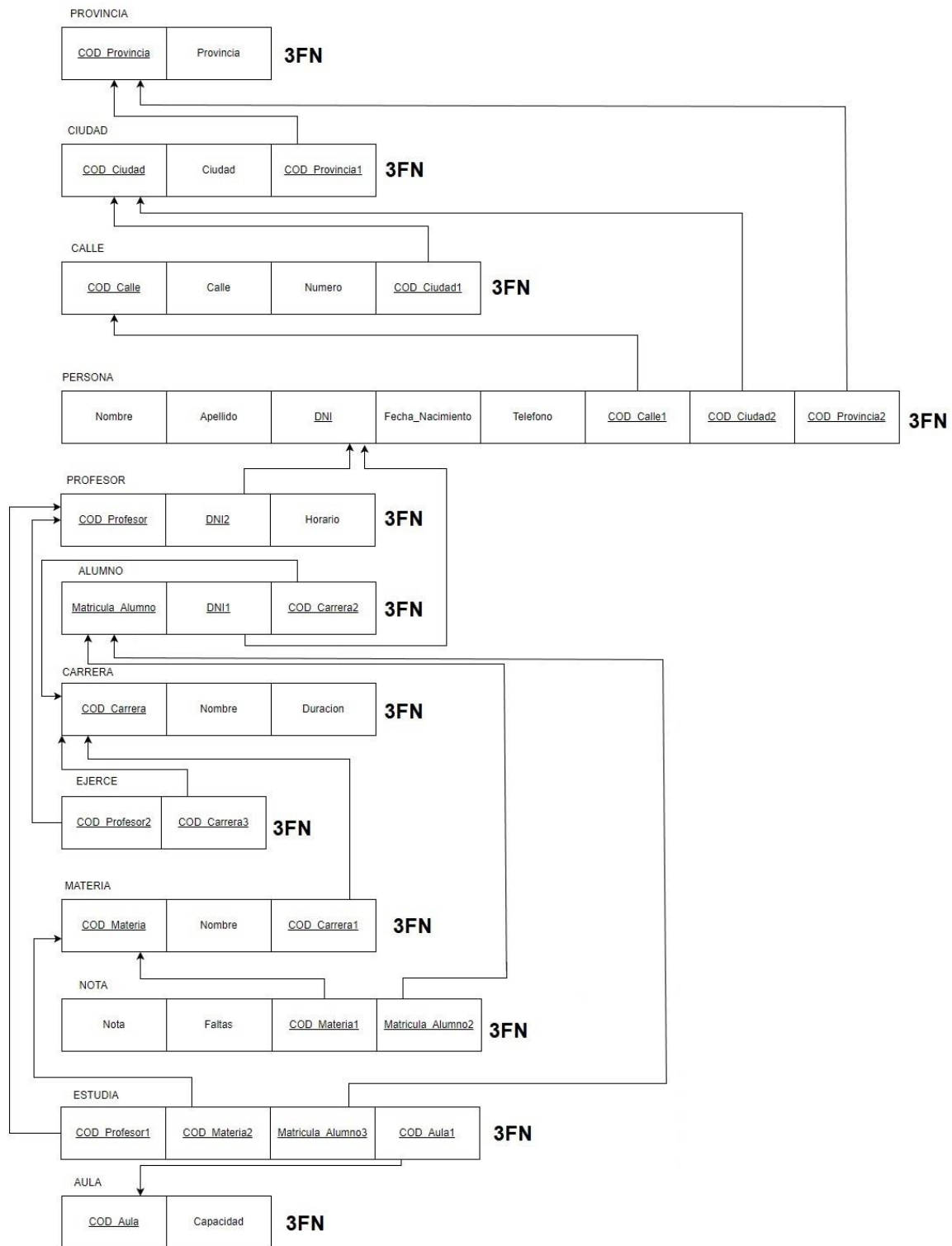
## **Desventajas.**

- *Si se quiere migrar una base de datos de MySQL 5.1 a MariaDB, esta tiene que ser la misma versión “MariaDB 5.1”.*

## Modelo E-R.



## **Modelo Relacional y Dependencias.**



## **Algebra Relacional.**

### **1 - Ver las materias que tiene la carrera (COD Carrera=1).**

$\pi(\text{ALUMNO.Matricula\_Alumno}, \text{PERSONA.Nombre}, \text{PERSONA.Apellido}) (\sigma(\text{MATERIA.COD\_Carrera1} = \text{CARRERA.COD\_Carrera} \wedge \text{COD\_Carrera} = 1)) (\text{MATERIA} \times \text{CARRERA})$

### **2 - Listar matricula, nombre, apellido y edad de los alumnos cuya edad sea mayor o igual a 25 años.**

$\pi(\text{ALUMNO.Matricula\_Alumno}, \text{PERSONA.Nombre}, \text{PERSONA.apellido}, \text{TIMESTAMPDIFF}(\text{YEAR}, \text{PERSONA.Fecha\_Nacimiento}, \text{CURDATE}()) \text{ AS 'Edad'}) (\sigma(\text{PERSONA.DNI} = \text{ALUMNO.DNI1} \wedge \text{WHERE } \text{TIMESTAMPDIFF}(\text{YEAR}, \text{PERSONA.Fecha\_Nacimiento}, \text{CURDATE}()) \geq 25 \text{ GROUP BY } \text{ALUMNO.Matricula\_Alumno})) (\text{PERSONA} \times \text{ALUMNO})$

### **3 - Buscar todos los datos personales del Alumno y que carrera estudia donde el DNI del alumno es 40716939**

$\pi(\text{PERSONA.Nombre}, \text{PERSONA.Apellido}, \text{ALUMNO.Matricula\_Alumno} \text{ AS 'Matricula'}, \text{PERSONA.Fecha\_Nacimiento} \text{ AS 'Fecha de nacimiento'}, \text{PERSONA.DNI}, \text{PERSONA.Telefono}, \text{PROVINCIA.Provincia}, \text{CIUDAD.Ciudad}, \text{CALLE.Calle}, \text{CALLE.Numero}, \text{CARRERA.Nombre} \text{ AS 'Carrera'}) (\sigma(\text{PERSONA.COD\_Provincia2} = \text{PROVINCIA.COD\_Provincia} \wedge \text{COD\_Provincia} = 1 \wedge \text{PERSONA.COD\_Ciudad2} = \text{CIUDAD.COD\_Ciudad} \wedge \text{COD\_Ciudad} = 1 \wedge \text{PERSONA.COD\_Calle1} = \text{CALLE.COD\_Calle} \wedge \text{COD\_Calle} = 1 \wedge \text{PERSONA.DNI} = \text{ALUMNO.DNI1} \wedge \text{CARRERA.COD\_Carrera} = \text{ALUMNO.COD\_Carrera2} \wedge \text{ALUMNO.DNI1} = 40716939)) (\text{PERSONA} \times \text{PROVINCIA} \times \text{Ciudad} \times \text{CALLE} \times \text{ALUMNO} \times \text{CARRERA})$

**4 - Buscar todos los datos personales del profesor y que carrera ejerce donde el DNI del profesor es 33333333**

$\pi$ (PERSONA.Nombre,  
PERSONA.Apellido, PROFESOR.COD\_Profesor,  
PERSONA.Fecha\_Nacimiento AS 'Fecha de nacimiento',  
PERSONA.DNI, PERSONA.Telefono, PROVINCIA.Provincia,  
CIUDAD.Ciudad, CALLE.Calle, CALLE.Numero, CARRERA.Nombre  
AS 'Carrera', PROFESOR.Horario)(  
 $\sigma$ (PERSONA.COD\_Provincia2=PROVINCIA.COD\_Provincia^PERSONA.COD\_Ciudad2=CIUDAD.COD\_Ciudad^PERSONA.COD\_Calle1=CALLE.COD\_Calle^PERSONA.DNI=PROFESOR.DNI2^PROFESOR.COD\_Profesor=EJERCE.COD\_Profesor2^PROFESOR.COD\_Carrera3=CARRERA.COD\_Carrera3^PROFESOR.DNI=33333333))  
(PERSONAxPROVINCIAxCIUDADxCALLExPROFESORxEJERCExCARRERA)

**5 – Mostrar promedio de Alumno en todas las materias y sus faltas donde la matricula del alumno es 1**

$\pi$ (ALUMNO.Matricula\_Alumno, MATERIA.Nombre, SUM(nota.faltas) AS 'Faltas', AVG(nota) AS 'Nota Final')  
( $\sigma$ (NOTA.Matricula\_Aumno2=ALUMNO.Matricula\_Alumno^NOTA.COD\_Materia1=MATERIA.COD\_Materia^ALUMNO.Matricula\_Alumno=1  
GROUP BY MATERIA.Nombre))  
(NOTAxAlumnoxMATERIA)

## **Definición de tablas.**

### **carrera.**

```
CREATE TABLE `carrera` (  
  `COD_Carrera` TINYINT(4) NOT NULL,  
  `Nombre` VARCHAR(60) NOT NULL COLLATE 'utf8mb4_general_ci',  
  `Duracion` FLOAT NULL DEFAULT NULL,  
  PRIMARY KEY(`COD_Carrera`) USING BTREE  
)
```

### **materia.**

```
CREATE TABLE `materia` (  
  `COD_Materia` INT(11) NOT NULL,  
  `Nombre` VARCHAR(50) NOT NULL COLLATE 'utf8mb4_general_ci',  
  `COD_Carrera1` TINYINT(4) NULL,  
  PRIMARY KEY (`COD_Materia`) USING BTREE,  
  INDEX `FK_COD_Carrera1` (`COD_Carrera1`) USING BTREE,  
  CONSTRAINT `FK_COD_Carrera1` FOREIGN KEY (`COD_Carrera1`)  
REFERENCES `proyecto`.`carrera` (`COD_Carrera`) ON UPDATE CASCADE ON  
DELETE CASCADE  
)
```

### **aula.**

```
CREATE TABLE `aula` (  
  `COD_Aula` TINYINT(4) NOT NULL,  
  `Capacidad` TINYINT(4) NOT NULL,  
  PRIMARY KEY (`COD_Aula`) USING BTREE,  
)
```

### **provincia.**

```
CREATE TABLE `provincia` (  
  `COD_Provincia` TINYINT(4) NOT NULL,  
  `Provincia` VARCHAR(51) NULL DEFAULT NULL COLLATE  
'utf8mb4_general_ci',  
  PRIMARY KEY (`COD_Provincia`) USING BTREE  
)
```



## **ciudad.**

```
CREATE TABLE `ciudad` (  
  `COD_Ciudad` INT(11) NOT NULL,  
  `Ciudad` VARCHAR(60) NOT NULL COLLATE 'utf8mb4_general_ci',  
  `COD_Provincial` TINYINT(4) NOT NULL,  
  PRIMARY KEY (`COD_Ciudad`) USING BTREE,  
  INDEX `FK_COD_Provincial` (`COD_Provincial`) USING BTREE,  
  CONSTRAINT `FK_COD_Provincial` FOREIGN KEY (`COD_Provincial`)  
REFERENCES `proyecto`.`provincia` (`COD_Provincia`) ON UPDATE CASCADE ON  
DELETE CASCADE  
)
```

## **calle.**

```
CREATE TABLE `calle` (  
  `COD_Calle` INT(11) NOT NULL,  
  `Calle` VARCHAR(60) NOT NULL COLLATE 'utf8mb4_general_ci',  
  `Numero` SMALLINT(6) NOT NULL,  
  `COD_Ciudad1` INT(11) NOT NULL,  
  PRIMARY KEY (`COD_Calle`) USING BTREE,  
  INDEX `FK_COD_Ciudad1` (`COD_Ciudad1`) USING BTREE,  
  CONSTRAINT `FK_COD_Ciudad1` FOREIGN KEY (`COD_Ciudad1`) REFERENCES  
`proyecto`.`ciudad` (`COD_Ciudad`) ON UPDATE CASCADE ON DELETE CASCADE  
)
```

## **persona.**

```
CREATE TABLE `persona` (  
  `Nombre` VARCHAR(60) NOT NULL COLLATE 'utf8mb4_general_ci',  
  `Apellido` VARCHAR(60) NOT NULL COLLATE 'utf8mb4_general_ci',  
  `Fecha_Nacimiento` DATE NULL,  
  `DNI` BIGINT(20) NOT NULL,  
  `Telefono` BIGINT(20) NOT NULL,  
  `COD_Provincia2` TINYINT(4) NOT NULL,  
  `COD_Calle1` INT(11) NOT NULL,  
  `COD_Ciudad2` INT(11) NOT NULL,  
  PRIMARY KEY (`DNI`) USING BTREE,  
  INDEX `FK_COD_Provincia2` (`COD_Provincia2`) USING BTREE,  
  INDEX `FK_COD_Calle1` (`COD_Calle1`) USING BTREE,  
  INDEX `FK_COD_Ciudad2` (`COD_Ciudad2`) USING BTREE,  
  CONSTRAINT `FK_COD_Calle1` FOREIGN KEY (`COD_Calle1`) REFERENCES  
`proyecto`.`calle` (`COD_Calle`) ON UPDATE NO ACTION ON DELETE NO ACTION,  
  CONSTRAINT `FK_COD_Ciudad2` FOREIGN KEY (`COD_Ciudad2`) REFERENCES  
`proyecto`.`ciudad` (`COD_Ciudad`) ON UPDATE NO ACTION ON DELETE NO  
ACTION,  
  CONSTRAINT `FK_COD_Provincia2` FOREIGN KEY (`COD_Provincia2`)  
REFERENCES `proyecto`.`provincia` (`COD_Provincia`) ON UPDATE NO ACTION  
ON DELETE NO ACTION  
)
```

## **profesor.**

```
CREATE TABLE `profesor` (  
  `COD_Profesor` INT(11) NOT NULL,  
  `Horario` VARCHAR(10) NULL DEFAULT NULL COLLATE  
  'utf8mb4_general_ci',  
  `DNI2` BIGINT(20) NOT NULL,  
  PRIMARY KEY (`COD_Profesor`) USING BTREE,  
  INDEX `FK_DNI2` (`DNI2`) USING BTREE,  
  CONSTRAINT `FK_DNI2` FOREIGN KEY (`DNI2`) REFERENCES  
  `proyecto`.`persona` (`DNI`) ON UPDATE CASCADE ON DELETE CASCADE  
)
```

## **alumno.**

```
CREATE TABLE `alumno` (  
  `Matricula_Alumno` INT(11) NOT NULL,  
  `DNI1` BIGINT(20) NOT NULL,  
  `COD_Carrera2` TINYINT(4) NOT NULL,  
  PRIMARY KEY (`Matricula_Alumno`) USING BTREE,  
  INDEX `FK_DNI1` (`DNI1`) USING BTREE,  
  INDEX `FK_COD_Carrera2` (`COD_Carrera2`) USING BTREE,  
  CONSTRAINT `FK_COD_Carrera2` FOREIGN KEY (`COD_Carrera2`) REFERENCES  
  `proyecto`.`carrera` (`COD_Carrera`) ON UPDATE CASCADE ON  
  DELETE CASCADE,  
  CONSTRAINT `FK_DNI1` FOREIGN KEY (`DNI1`) REFERENCES  
  `proyecto`.`persona` (`DNI`) ON UPDATE CASCADE ON DELETE CASCADE  
)
```

## **nota.**

```
CREATE TABLE `nota` (  
  `Nota` FLOAT NULL DEFAULT NULL,  
  `faltas` TINYINT(4) NOT NULL,  
  `Matricula_Alumno2` INT(11) NOT NULL,  
  `COD_Material` INT(11) NOT NULL,  
  INDEX `FK_Matricula_Alumno2` (`Matricula_Alumno2`) USING BTREE,  
  INDEX `FK_COD_Material` (`COD_Material`) USING BTREE,  
  CONSTRAINT `FK_COD_Material` FOREIGN KEY (`COD_Material`) REFERENCES  
  `proyecto`.`materia` (`COD_Materia`) ON UPDATE CASCADE ON  
  DELETE CASCADE,  
  CONSTRAINT `FK_Matricula_Alumno2` FOREIGN KEY (`Matricula_Alumno2`) REFERENCES  
  `proyecto`.`alumno` (`Matricula_Alumno`) ON UPDATE CASCADE ON  
  DELETE CASCADE  
)
```

## **estudia.**

```
CREATE TABLE `estudia` (  
  `COD_Profesor1` INT(11) NOT NULL,  
  `COD_Materia2` INT(11) NOT NULL,  
  `Matricula_Alumno3` INT(11) NOT NULL,  
  `COD_Aula1` TINYINT(4) NOT NULL,  
  INDEX `FK_COD_Profesor1` (`COD_Profesor1`) USING BTREE,  
  INDEX `FK_COD_Materia2` (`COD_Materia2`) USING BTREE,  
  INDEX `FK_Matricula_Alumno3` (`Matricula_Alumno3`) USING BTREE,  
  INDEX `FK_COD_Aula1` (`COD_Aula1`) USING BTREE,  
  CONSTRAINT `FK_COD_Aula1` FOREIGN KEY (`COD_Aula1`) REFERENCES  
`proyecto`.`aula` (`COD_Aula`) ON UPDATE CASCADE ON DELETE CASCADE,  
  CONSTRAINT `FK_COD_Materia2` FOREIGN KEY (`COD_Materia2`) REFERENCES  
`proyecto`.`materia` (`COD_Materia`) ON UPDATE CASCADE ON  
DELETE CASCADE,  
  CONSTRAINT `FK_COD_Profesor1` FOREIGN KEY (`COD_Profesor1`) REFERENCES  
`proyecto`.`profesor` (`COD_Profesor`) ON UPDATE CASCADE ON  
DELETE CASCADE,  
  CONSTRAINT `FK_Matricula_Alumno3` FOREIGN KEY (`Matricula_Alumno3`) REFERENCES  
`proyecto`.`alumno` (`Matricula_Alumno`) ON UPDATE CASCADE ON  
DELETE CASCADE  
)
```

## **ejerce.**

```
CREATE TABLE `ejerce` (  
  `COD_Profesor2` INT(11) NOT NULL,  
  `COD_Carrera3` TINYINT(4) NOT NULL,  
  INDEX `FK_COD_Profesor2` (`COD_Profesor2`) USING BTREE,  
  INDEX `FK_COD_Carrera3` (`COD_Carrera3`) USING BTREE,  
  CONSTRAINT `FK_COD_Carrera3` FOREIGN KEY (`COD_Carrera3`) REFERENCES  
`proyecto`.`carrera` (`COD_Carrera`) ON UPDATE CASCADE ON  
DELETE CASCADE,  
  CONSTRAINT `FK_COD_Profesor2` FOREIGN KEY (`COD_Profesor2`) REFERENCES  
`proyecto`.`profesor` (`COD_Profesor`) ON UPDATE CASCADE ON  
DELETE CASCADE)
```

## **Carga de datos.**

### **carrera.**

```
INSERT INTO `proyecto`.`carrera` (`COD_Carrera`, `Nombre`, `Duracion`)
VALUES ('1', 'Tecnicatura en programacion', '2');

INSERT INTO `proyecto`.`carrera` (`COD_Carrera`, `Nombre`, `Duracion`)
VALUES ('2', 'Seguridad e higiene', '2');

INSERT INTO `proyecto`.`carrera` (`COD_Carrera`, `Nombre`, `Duracion`)
VALUES ('3', 'Licenciatura en Nutrición', '3');

INSERT INTO `proyecto`.`carrera` (`COD_Carrera`, `Nombre`, `Duracion`)
VALUES ('4', 'Diplomatura en Diseño de interiores', '2');

INSERT INTO `proyecto`.`carrera` (`COD_Carrera`, `Nombre`, `Duracion`)
VALUES ('5', 'Diplomatura Universitaria en Producción Audiovisual', '2');
```

### **materia.**

```
INSERT INTO `proyecto`.`materia` (`COD_Materia`, `Nombre`,
`COD_Carrera1`) VALUES ('1', 'Diseño y administracion de base de
datos', '1');

INSERT INTO `proyecto`.`materia` (`COD_Materia`, `Nombre`,
`COD_Carrera1`) VALUES ('2', 'Laboratorio II', '1');

INSERT INTO `proyecto`.`materia` (`COD_Materia`, `Nombre`,
`COD_Carrera1`) VALUES ('3', 'Seguridad', '2');

INSERT INTO `proyecto`.`materia` (`COD_Materia`, `Nombre`,
`COD_Carrera1`) VALUES ('4', 'Metología en sistemas', '1');

INSERT INTO `proyecto`.`materia` (`COD_Materia`, `Nombre`,
`COD_Carrera1`) VALUES ('5', 'Legislación', '1');

INSERT INTO `proyecto`.`materia` (`COD_Materia`, `Nombre`,
`COD_Carrera1`) VALUES ('6', 'Laboratorio IV', '1');
```

### **aula.**

```
INSERT INTO `proyecto`.`aula` (`COD_Aula`, `Capacidad`) VALUES ('1',
'45');

INSERT INTO `proyecto`.`aula` (`COD_Aula`, `Capacidad`) VALUES ('2',
'35');
```

## **provincia.**

```
INSERT INTO `proyecto`.`provincia` (`COD_Provincia`, `Provincia`) VALUES ('1', 'Buenos Aires');
```

## **ciudad.**

```
INSERT INTO `proyecto`.`ciudad` (`COD_Ciudad`, `Ciudad`, `COD_Provincial`) VALUES ('1', 'Chivilcoy', '1');
```

```
INSERT INTO `proyecto`.`ciudad` (`COD_Ciudad`, `Ciudad`, `COD_Provincial`) VALUES ('2', 'Buenos Aires', '1');
```

```
INSERT INTO `proyecto`.`ciudad` (`COD_Ciudad`, `Ciudad`, `COD_Provincial`) VALUES ('3', 'Chacabuco', '1');
```

## **calle.**

```
INSERT INTO `proyecto`.`calle` (`COD_Calle`, `Calle`, `Numero`, `COD_Ciudad1`) VALUES ('1', 'Bomberos Voluntarios', '101', '1');
```

```
INSERT INTO `proyecto`.`calle` (`COD_Calle`, `Calle`, `Numero`, `COD_Ciudad1`) VALUES ('2', 'Pellegrini', '115', '2');
```

```
INSERT INTO `proyecto`.`calle` (`COD_Calle`, `Calle`, `Numero`, `COD_Ciudad1`) VALUES ('3', 'Roseti', '250', '1');
```

```
INSERT INTO `proyecto`.`calle` (`COD_Calle`, `Calle`, `Numero`, `COD_Ciudad1`) VALUES ('4', 'Bomberos Voluntarios', '110', '1');
```

```
INSERT INTO `proyecto`.`calle` (`COD_Calle`, `Calle`, `Numero`, `COD_Ciudad1`) VALUES ('5', 'Laprida', '50', '3');
```

```
INSERT INTO `proyecto`.`calle` (`COD_Calle`, `Calle`, `Numero`, `COD_Ciudad1`) VALUES ('6', 'Alvear', '25', '3');
```

## **persona.**

```
INSERT INTO `proyecto`.`persona` (`Nombre`, `Apellido`, `Fecha_Nacimiento`, `DNI`, `Telefono`, `COD_Provincia2`, `COD_Calle1`, `COD_Ciudad2`) VALUES ('Agustin', 'Narbeburý', '1997-08-13', '40716939', '2346511650', '1', '1', '2');
```

```
INSERT INTO `proyecto`.`persona` (`Nombre`, `Apellido`, `Fecha_Nacimiento`, `DNI`, `Telefono`, `COD_Provincia2`, `COD_Calle1`, `COD_Ciudad2`) VALUES ('Lucas', 'Mesas', '1970-10-31', '33333333', '2346555666', '1', '2', '2');
```

```
INSERT INTO `proyecto`.`persona` (`Nombre`, `Apellido`,
`Fecha_Nacimiento`, `DNI`, `Telefono`, `COD_Provincia2`, `COD_Calle1`,
`COD_Ciudad2`) VALUES ('Paco', 'Montoya', '1996-11-08', '2222222',
'2346666333', '1', '3', '1');
```

```
INSERT INTO `proyecto`.`persona` (`Nombre`, `Apellido`,
`Fecha_Nacimiento`, `DNI`, `Telefono`, `COD_Provincia2`, `COD_Calle1`,
`COD_Ciudad2`) VALUES ('Juan', 'Perez', '2000-01-01', '11111111',
'2346666666', '1', '3', '1');
```

```
INSERT INTO `proyecto`.`persona` (`Nombre`, `Apellido`,
`Fecha_Nacimiento`, `DNI`, `Telefono`, `COD_Provincia2`, `COD_Calle1`,
`COD_Ciudad2`) VALUES ('Marcelo', 'De Lillo', '1981-11-10', '22222221',
'234622221', '1', '4', '1');
```

```
INSERT INTO `proyecto`.`persona` (`Nombre`, `Apellido`,
`Fecha_Nacimiento`, `DNI`, `Telefono`, `COD_Provincia2`, `COD_Calle1`,
`COD_Ciudad2`) VALUES ('Edith', 'Tabella', '1967-11-05', '22222211',
'2344333111', '1', '5', '3');
```

```
INSERT INTO `proyecto`.`persona` (`Nombre`, `Apellido`,
`Fecha_Nacimiento`, `DNI`, `Telefono`, `COD_Provincia2`, `COD_Calle1`,
`COD_Ciudad2`) VALUES ('Roxana', 'Tange', '1978-11-10', '33333332',
'222222212', '1', '6', '3');
```

## **alumno.**

```
INSERT INTO `proyecto`.`alumno` (`Matricula_Alumno`, `DNI1`,
`COD_Carrera2`) VALUES ('1', '40716939', '1');
```

```
INSERT INTO `proyecto`.`alumno` (`Matricula_Alumno`, `DNI1`,
`COD_Carrera2`) VALUES ('2', '11111111', '2');
```

```
INSERT INTO `proyecto`.`alumno` (`Matricula_Alumno`, `DNI1`,
`COD_Carrera2`) VALUES ('3', '22222222', '1');
```

## **profesor.**

```
INSERT INTO `proyecto`.`profesor` (`COD_Profesor`, `Horario`, `DNI2`)
VALUES ('1', 'mixto', '33333333');
```

```
INSERT INTO `proyecto`.`profesor` (`COD_Profesor`, `Horario`, `DNI2`)
VALUES ('2', 'vespertino', '2222221');
```

```
INSERT INTO `proyecto`.`profesor` (`COD_Profesor`, `Horario`, `DNI2`)
VALUES ('3', 'mixto', '22222211');
```

```
INSERT INTO `proyecto`.`profesor` (`COD_Profesor`, `Horario`, `DNI2`)
VALUES ('4', 'vespertino', '33333332');
```

## **nota.**

```
INSERT INTO `proyecto`.`nota` (`Nota`, `Faltas`, `Matricula_Alumno2`,  
`COD_Material`) VALUES ('10', '0', '1', '1');
```

```
INSERT INTO `proyecto`.`nota` (`Nota`, `Faltas`, `Matricula_Alumno2`,  
`COD_Material`) VALUES ('8', '1', '1', '1');
```

```
INSERT INTO `proyecto`.`nota` (`Nota`, `Faltas`, `Matricula_Alumno2`,  
`COD_Material`) VALUES ('8', '0', '1', '2');
```

```
INSERT INTO `proyecto`.`nota` (`Nota`, `Faltas`, `Matricula_Alumno2`,  
`COD_Material`) VALUES ('8', '0', '1', '2');
```

```
INSERT INTO `proyecto`.`nota` (`Nota`, `Faltas`, `Matricula_Alumno2`,  
`COD_Material`) VALUES ('8', '0', '1', '2');
```

```
INSERT INTO `proyecto`.`nota` (`Nota`, `Faltas`, `Matricula_Alumno2`,  
`COD_Material`) VALUES ('8', '0', '1', '2');
```

```
INSERT INTO `proyecto`.`nota` (`Nota`, `Faltas`, `Matricula_Alumno2`,  
`COD_Material`) VALUES ('8', '0', '1', '4');
```

```
INSERT INTO `proyecto`.`nota` (`Nota`, `Faltas`, `Matricula_Alumno2`,  
`COD_Material`) VALUES ('8', '0', '1', '4');
```

## **estudia.**

```
INSERT INTO `proyecto`.`estudia` (`COD_Profesor1`, `COD_Materia2`,  
`Matricula_Alumno3`, `COD_Aula1`) VALUES ('1', '1', '1', '1');
```

```
INSERT INTO `proyecto`.`estudia` (`COD_Profesor1`, `COD_Materia2`,  
`Matricula_Alumno3`, `COD_Aula1`) VALUES ('2', '2', '1', '1');
```

```
INSERT INTO `proyecto`.`estudia` (`COD_Profesor1`, `COD_Materia2`,  
`Matricula_Alumno3`, `COD_Aula1`) VALUES ('2', '6', '1', '1');
```

```
INSERT INTO `proyecto`.`estudia` (`COD_Profesor1`, `COD_Materia2`,  
`Matricula_Alumno3`, `COD_Aula1`) VALUES ('3', '5', '1', '1');
```

```
INSERT INTO `proyecto`.`estudia` (`COD_Profesor1`, `COD_Materia2`,  
`Matricula_Alumno3`, `COD_Aula1`) VALUES ('1', '1', '2', '1');
```

```
INSERT INTO `proyecto`.`estudia` (`COD_Profesor1`, `COD_Materia2`,  
`Matricula_Alumno3`, `COD_Aula1`) VALUES ('1', '1', '3', '1');
```

```
INSERT INTO `proyecto`.`estudia` (`COD_Profesor1`, `COD_Materia2`,  
`Matricula_Alumno3`, `COD_Aula1`) VALUES ('4', '4', '1', '1');
```

```
INSERT INTO `proyecto`.`estudia` (`COD_Profesor1`, `COD_Materia2`,  
`Matricula_Alumno3`, `COD_Aula1`) VALUES ('2', '6', '2', '1');
```

## **ejerce.**

```
INSERT INTO `proyecto`.`ejerce` (`COD_Profesor2`, `COD_Carrera3`) VALUES ('1', '1');
```

```
INSERT INTO `proyecto`.`ejerce` (`COD_Profesor2`, `COD_Carrera3`) VALUES ('2', '1');
```

```
INSERT INTO `proyecto`.`ejerce` (`COD_Profesor2`, `COD_Carrera3`) VALUES ('3', '1');
```

```
INSERT INTO `proyecto`.`ejerce` (`COD_Profesor2`, `COD_Carrera3`) VALUES ('4', '1');
```

## **Consultas SQL.**

### **1 - Ver las materias que tiene la carrera donde el COD Carrera = 1.**

```
SELECT materia.Nombre, carrera.Nombre FROM materia, carrera WHERE materia.COD_Carrera1=carrera.COD_Carrera AND carrera.COD_Carrera=1
```

#### **Dato Obtenido con la consulta:**



Nombre	Nombre
Diseño y administración de base de datos	Tecnatura en programación
Laboratorio II	Tecnatura en programación
Metodología en sistemas	Tecnatura en programación
Legislación	Tecnatura en programación
Laboratorio IV	Tecnatura en programación

### **2 - Listar matricula, nombre, apellido y edad de los alumnos cuya edad sea mayor o igual a 25 años.**

```
SELECT alumno.Matricula_Alumno, persona.Nombre, persona.Apellido, TIMESTAMPDIFF(YEAR, persona.Fecha_Nacimiento, CURDATE()) AS 'Edad' FROM alumno INNER JOIN persona ON alumno.DNI1=persona.DNI WHERE TIMESTAMPDIFF(YEAR, persona.Fecha_Nacimiento, CURDATE())>=25 GROUP BY alumno.Matricula_Alumno;
```

#### **Dato Obtenido con la consulta:**



Matricula_Alumno	Nombre	Apellido	Edad
1	Agustin	Narbebury	25
3	Paco	Montoya	26



### **3 - Buscar todos los datos personales del alumno y que carrera estudia donde el DNI del alumno es 40716939**

```
SELECT persona.Nombre, persona.Apellido, alumno.Matricula_Alumno AS
'Matricula', persona.Fecha_Nacimiento AS 'Fecha de
nacimiento', persona.DNI, persona.Telefono, provincia.Provincia,
ciudad.Ciudad, calle.Calle, calle.Numero, carrera.Nombre AS 'Carrera'
FROM persona INNER JOIN provincia ON
persona.COD_Provincia2=provincia.COD_Provincia
INNER JOIN ciudad ON persona.COD_Ciudad2=ciudad.COD_Ciudad
INNER JOIN calle ON persona.COD_Calle1=calle.COD_Calle
INNER JOIN alumno ON persona.DNI=alumno.DNI1
INNER JOIN carrera ON carrera.COD_Carrera=alumno.COD_Carrera2 WHERE
alumno.DNI1 = 40716939
```

#### **Dato Obtenido con la consulta:**



The screenshot shows a database query result with 11 columns: Nombre, Apellido, Matricula, Fecha de nacimiento, DNI, Telefono, Provincia, Ciudad, Calle, Numero, and Carrera. The data row shows: Agustin, Narbebury, 1, 1997-08-13, 40.716.939, 2.346.511.650, Buenos Aires, Chivilcoy, Bomberos Voluntarios, 101, and Tecnicatura en programacion. The status bar at the bottom indicates 'Connected: 2 days, 18:43 h', 'MariaDB 10.8.3', 'Uptime: 4 days, 17:38 h', and 'Server time: 11:28'.

Nombre	Apellido	Matricula	Fecha de nacimiento	DNI	Telefono	Provincia	Ciudad	Calle	Numero	Carrera
Agustin	Narbebury	1	1997-08-13	40.716.939	2.346.511.650	Buenos Aires	Chivilcoy	Bomberos Voluntarios	101	Tecnicatura en programacion

### **4 - Buscar todos los datos personales del profesor y que carrera ejerce donde el DNI del profesor es 33333333**

```
SELECT
persona.Nombre, persona.Apellido, profesor.COD_Profesor, persona.Fecha_Nacim
iento AS 'Fecha de nacimiento', persona.DNI, persona.Telefono,
provincia.Provincia, ciudad.Ciudad, calle.Calle, calle.Numero,
carrera.Nombre AS 'Carrera', profesor.Horario FROM persona INNER JOIN
provincia ON persona.COD_Provincia2=provincia.COD_Provincia
INNER JOIN ciudad ON persona.COD_Ciudad2=ciudad.COD_Ciudad
INNER JOIN calle ON persona.COD_Calle1=calle.COD_Calle
INNER JOIN profesor ON persona.DNI=profesor.DNI2
INNER JOIN ejerce ON profesor.COD_Profesor=ejerce.COD_Profesor2
INNER JOIN carrera ON ejerce.COD_Carrera3=carrera.COD_Carrera WHERE
profesor.DNI2 = 33333333;
```

#### **Dato Obtenido con la consulta:**



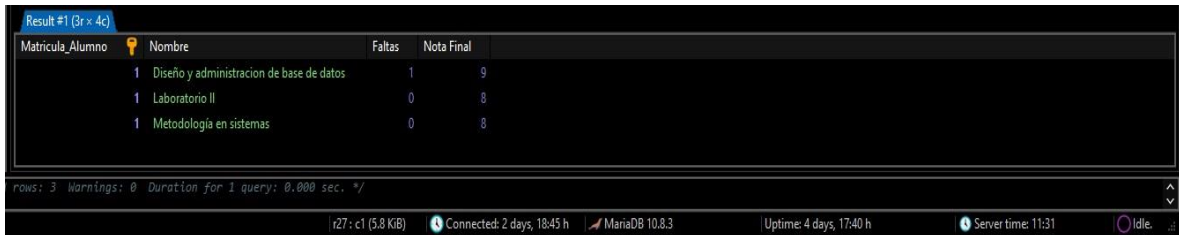
The screenshot shows a database query result with 12 columns: Nombre, Apellido, COD\_Profesor, Fecha\_Nacimiento, DNI, Telefono, Provincia, Ciudad, Calle, Numero, Carrera, and Horario. The data row shows: Lucas, Mesas, 1, 1980-10-31, 33.333.333, 2.346.555.666, Buenos Aires, Buenos Aires, Pellegrini, 115, Tecnicatura en programacion, and mixto. The status bar at the bottom indicates 'Connected: 2 days, 18:29 h', 'MariaDB 10.8.3', 'Uptime: 4 days, 17:24 h', and 'Server time: 11:14'.

Nombre	Apellido	COD_Profesor	Fecha_Nacimiento	DNI	Telefono	Provincia	Ciudad	Calle	Numero	Carrera	Horario
Lucas	Mesas	1	1980-10-31	33.333.333	2.346.555.666	Buenos Aires	Buenos Aires	Pellegrini	115	Tecnicatura en programacion	mixto

## **5 – Mostrar nota final de Alumno en todas las materias y sus faltas donde la matricula del alumno es 1**

```
SELECT alumno.Matricula_Alumno, materia.Nombre, SUM(nota.faltas) AS  
'Faltas', AVG(nota) AS 'Nota Final' FROM nota INNER JOIN alumno ON  
nota.Matricula_Alumno2=alumno.Matricula_Alumno  
INNER JOIN materia ON nota.COD_Material=materia.COD_Materia  
WHERE alumno.Matricula_Alumno = 1 GROUP BY materia.Nombre
```

### **Dato Obtenido con la consulta:**



The screenshot shows a MySQL query result with 3 rows and 4 columns. The columns are Matricula\_Alumno, Nombre, Faltas, and Nota Final. The data is as follows:

Matricula_Alumno	Nombre	Faltas	Nota Final
1	Diseño y administracion de base de datos	1	9
1	Laboratorio II	0	8
1	Metodología en sistemas	0	8

Below the table, it says "rows: 3 Warnings: 0 Duration for 1 query: 0.000 sec. \*/". At the bottom, there is a status bar with information like "r27: c1 (5.8 KiB)", "Connected: 2 days, 18:45 h", "MariaDB 10.8.3", "Uptime: 4 days, 17:40 h", "Server time: 11:31", and "Idle".

## **Creación usuarios y asignación acceso a los mismos.**

### **Creación de usuarios.**

```
GRANT USAGE ON proyecto.* TO ADMIN@localhost IDENTIFIED BY 'ADMIN123';  
GRANT USAGE ON proyecto.nota TO Agustin@localhost IDENTIFIED BY 'Agus939';  
GRANT USAGE ON proyecto.nota TO Paco@localhost IDENTIFIED BY 'Paco33';
```

### **Agrego permisos en usuarios**

```
GRANT ALL PRIVILEGES ON proyecto.* TO ADMIN@localhost;  
GRANT SELECT ON proyecto.nota TO Agustin@localhost;  
GRANT SELECT ON proyecto.nota TO Paco@localhost;
```

### **Veo privilegios en usuarios:**

```
SELECT * FROM mysql.user;  
SHOW GRANTS FOR ADMIN@localhost;  
SHOW GRANTS FOR Agustin@localhost;  
SHOW GRANTS FOR Paco@localhost;
```