

BIG DATA Y PYTHON

MÁSTER EN BIG DATA

201020

GABRIEL MARÍN DÍAZ

hola

Presentación

Yo mismo

Nombre: Gabriel Marín Díaz

A qué me dedico...

- Channel Enablement Manager en Sage
- Profesor Asociado UCM

Perfil de LinkedIn: <https://www.linkedin.com/in/gabrielmarindiaz/>

Vosotros?

CONTENIDO

Contenido

Resumen

Tema 1 – Introducción a SQL

Tema 2 – Introducción al Lenguaje Python

Tema 3 – HTML y Python

Tema 4 – Big Data y Python

Tema 5 – Procesamiento Distribuido (Spark)

Prácticas las realizaremos con Python, MongoDB, Apache Spark

Arrancamos?

ÍNDICE

Índice

Contexto

- Diagramas Entidad / Relación
- Diagramas Relacionales
- Un repaso por SQL
- Un repaso por los Lenguajes de Programación

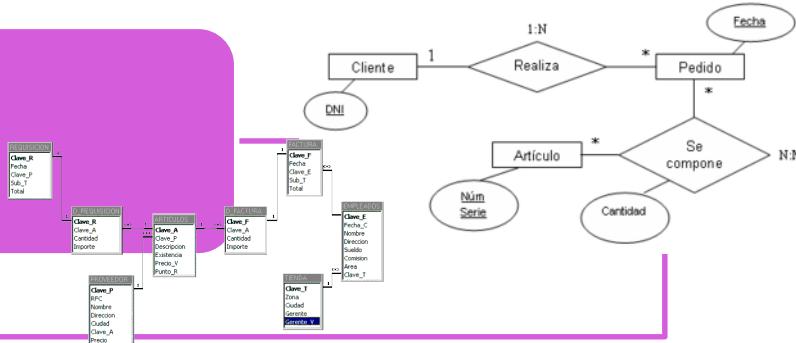
DIAGRAMAS ENTIDAD / RELACIÓN

Fases de diseño de una Base de Datos

Especificación de requisitos



Diseño conceptual



Implementación



ORACLE®



XAMPP

Fases de diseño de una Base de Datos

Especificación de requisitos del usuario, caracterizar completamente las necesidades de datos esperados por los usuarios de la base de datos.



Fases de diseño de una Base de Datos

Diseño conceptual

El diseñador traduce los requisitos a un esquema conceptual de base de datos

- Proporciona un **visión detallada** del desarrollo.
- El esquema especifica los conjuntos de **entidades**, conjuntos de **relaciones**, **atributos** y **ligaduras** de correspondencia.
- Se examina el diseño para **eliminar características redundantes**.
- Describir los datos y las relaciones, más que especificar detalles del almacenamiento físico.
- Especificación de **requisitos funcionales**. Serán especificados por un desarrollo del esquema conceptual.
- Los usuarios describen los tipos de **operación** transacciones que tendrán lugar en los datos.
- Las operaciones son, por ejemplo, la modificación o actualización de datos, la búsqueda y recuperación de datos específicos y el borrado de datos.

Fases de diseño de una Base de Datos

Fase de diseño lógico,

el esquema conceptual de alto nivel está asociado al **modelo de datos** de implementación de los SGBD.

Fase de diseño físico,

el esquema de la fase anterior se usa en esta, en la cual se especifican las **características físicas** de la base de datos.

Estas características incluyen la forma de organización de los archivos y las estructuras de almacenamiento interno.

Proceso de diseño: **Peligros**

1. **Redundancia**, la información debe aparecer en un solo lugar, no deberían existir duplicados. Consistencia de la información.
2. **Incompletitud**, aspectos mal modelados que redundan en la mala calidad del dato.
3. **Ambigüedad**, necesidad de identificación unívoca y no ambigua.
4. **Escalabilidad**, el sistema pueda ser escalable aplicando estrategias de gestión de la información, p.e. diccionario de datos, campos de usuario,...

Requisitos

**CÓMO PENSÁIS QUE SE HACE UNA TOMA
DE REQUISITOS?**

Requisitos

**EN LA MAYORÍA DE LOS CASOS EL CLIENTE
NO SABE LO QUE QUIERE**

Requisitos: Ejemplo

Requisitos necesarios para el diseño e implementación de un sistema relacional que permita consultar los alumnos matriculados en asignaturas así como el profesor que imparte las mismas:

- Un alumno puede estar matriculado en una o varias asignaturas.
- Además puede estar matriculado en la misma asignatura más de un curso escolar (si repite).
- Se quiere saber el curso escolar en el que cada alumno está matriculado de cada asignatura.
- En una asignatura habrá como mínimo 10 y como máximo 20 alumnos.
- Una asignatura es impartida por un único profesor.
- Un profesor podrá impartir varias asignaturas.

Mala calidad del dato: Ejercicio

CITAR EJEMPLOS DE EXPERIENCIA
DIARIA DONDE PENSÁIS QUE
PUDIERA EXISTIR UNA **MALA**
CALIDAD DEL DATO

Modelo Entidad-Relación

Los objetos en el mundo real se relacionan entre sí, la modelización de estas relaciones... (**SUJETO + VERBO + COMPLEMENTO**)

- El modelo está basado en que los conjuntos de datos forman **entidades** y existen **relaciones** entre esas entidades.
- Fácil de aprender y modelar en la mayoría de los casos.
- Existen herramientas software para ayuda en el diseño gráfico... en nuestro caso utilizaremos <https://www.mysql.com/products/workbench/>

Conceptos Básicos



Objetos Básicos = entidades

Relaciones entre los objetos

Una entidad

Cosa u objeto en el mundo real que es **distinguible** de todos los demás objetos → *cada persona*

Una entidad tiene un conjunto de **propiedades**, y los valores para algún conjunto de propiedades pueden identificar una entidad de forma única → *el DNI identifica únicamente a una persona*

La entidad puede ser **concreta** (*persona, libro*) o **abstracta** (*préstamo, vacaciones*)

Conceptos Básicos

Conjunto de entidades

- Todas las entidades del mismo tipo que comparten las mismas propiedades o atributos, *conjunto de todas las personas que son clientes del banco.*
- Los conjuntos de entidades no son necesariamente disjuntos, *una persona puede ser una entidad empleado, o una entidad cliente, ambas cosas, o ninguna.*
- Una **entidad** se representa por un *conjunto de atributos.*

Conjunto de entidades cliente y préstamo

32.112.312	Santos	Mayor	Peguerinos
1.928.374	Gómez	Carretas	Cerceda
67.789.901	López	Mayor	Peguerinos
55.555.555	Sotoca	Real	Cádiz
24.466.880	Pérez	Carretas	Cerceda
96.396.396	Valdivieso	Goya	Vigo
33.557.799	Fernández	Jazmín	León

cliente

P-17	1.000
P-23	2.000
P-15	1.500
P-14	1.500
P-19	500
P-11	900
P-16	1.300

préstamo

Conceptos Básicos: Atributos

- Describen las **propiedades** que posee cada miembro de un conjunto de entidades, p.e. *atributos del conjunto de entidades cliente: nombre, DNI, calle, ciudad.*
- Para cada atributo hay un **conjunto de valores** permitido, llamado **dominio**, el *dominio del atributo calle podría ser un conjunto de todas las cadenas de texto de un determinada longitud.*
- Un **atributo** de un conjunto de entidades es una función que asigna al conjunto de entidades un valor del dominio. **entidad = (atributo, valor)**
- Una **entidad** puede tener diferentes atributos, **Cliente = {(nombre, López), (DNI, 980789), (calle, Real), (ciudad, Segovia)}**.

Conceptos Básicos: Tipos de Atributos

- **Simples y compuestos**, el atributo *nombre* de un cliente se puede dividir en *nombre*, *primer-apellido* y *segundo-apellido*.
- **Univalueados y multivalueados**, en el conjunto de entidades empleado consideramos el atributo *nombre-subordinado*, cualquier empleado puede tener cero, uno, dos o más subordinados. O número de teléfono.
- **Nulos**, en *nombre-subordinado* si no tiene ningún subordinado.
- **Derivados**, el atributo *antigüedad* de un empleado dependerá del valor del atributo *fecha-comienzo* y de la fecha actual, *fecha-comienzo* será un atributo base o almacenado.



Conceptos Básicos: Relación

- Asociación entre diferentes entidades

Conjunto de relaciones

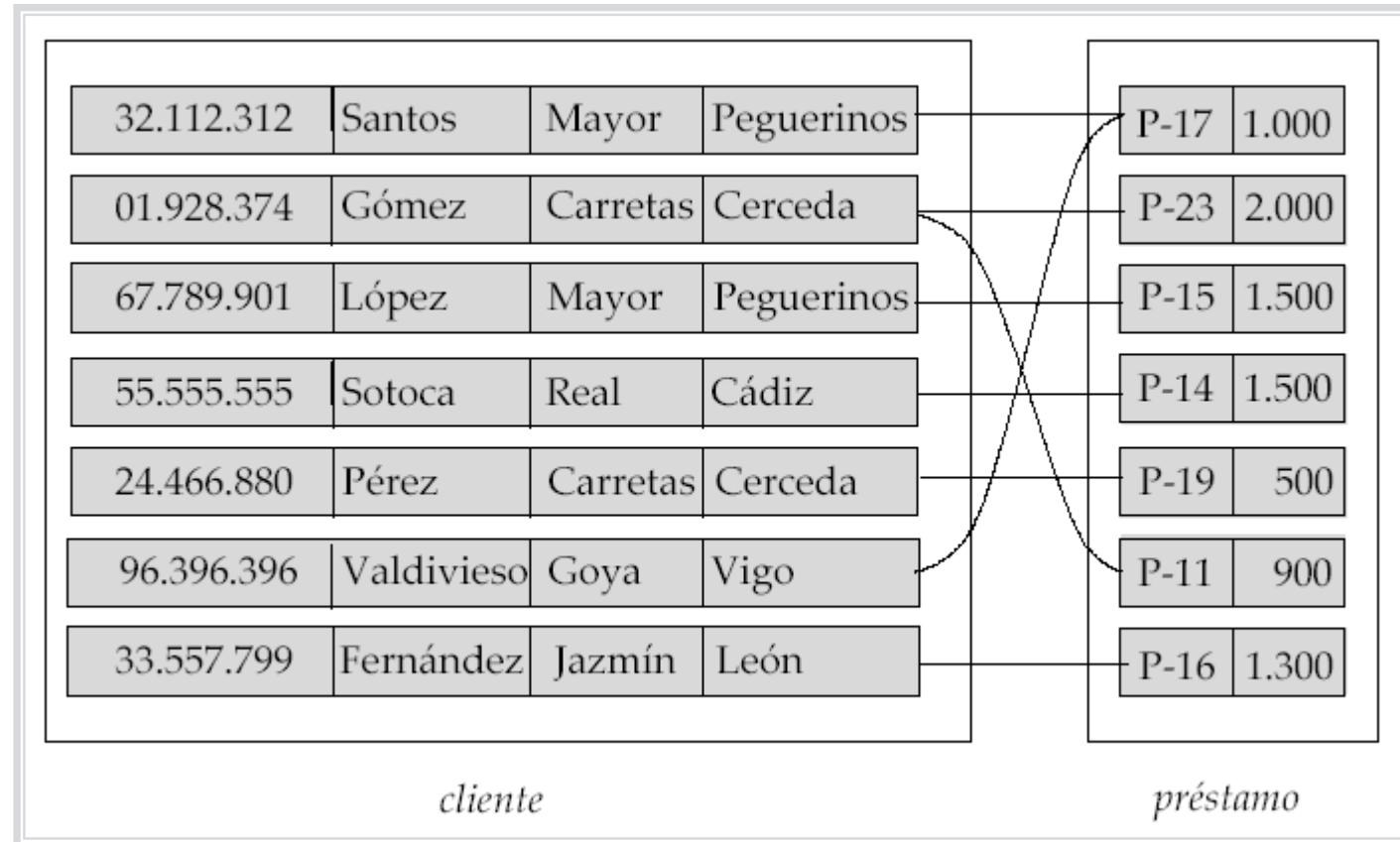
- Conjunto de relaciones del mismo tipo
- Formalmente es una relación matemática entre $n \geq 2$ entidades, cada una de ellas tomadas de los conjuntos de entidades

Si E_1, E_2, \dots, E_n son conjuntos de entidades, entonces un *conjunto de relaciones R* es un subconjunto de

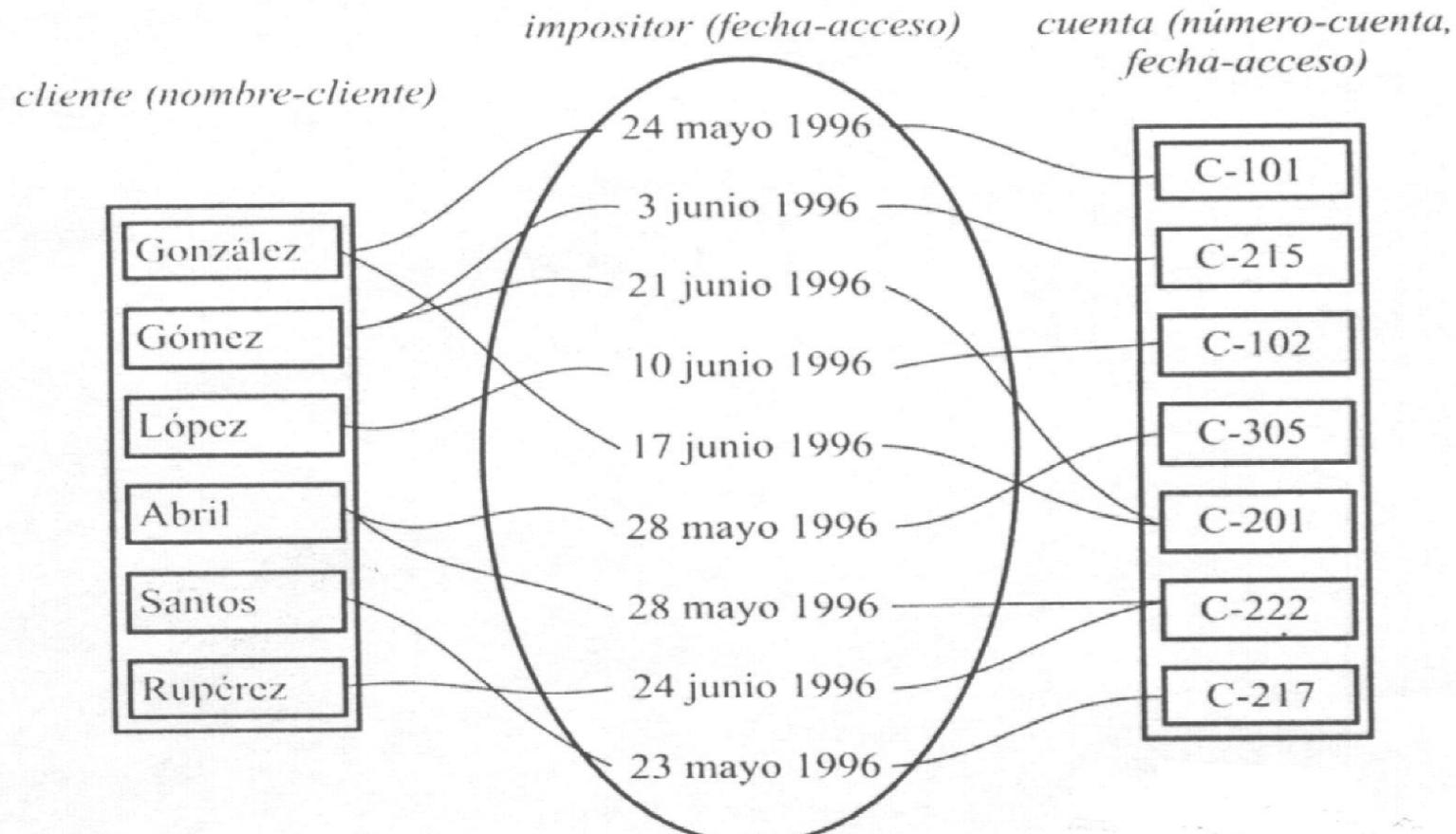
$$\{(e_1, e_2, \dots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n\}$$

donde (e_1, e_2, \dots, e_n) es una relación.

Conjunto de relaciones. Ejemplo



Conjunto de relaciones. Ejemplo



Ligaduras de correspondencia

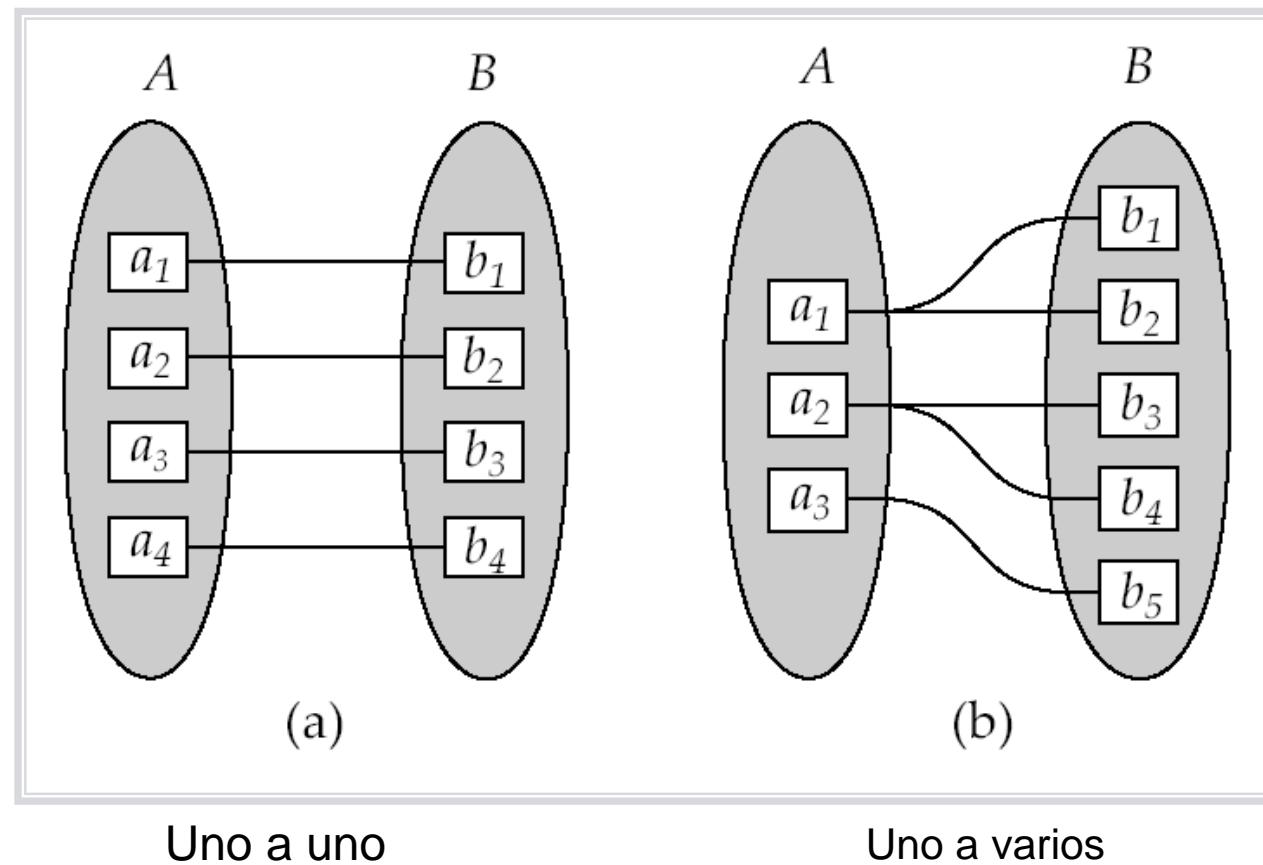
Ligaduras a las que la Base de Datos tiene que adaptarse

- Correspondencia de cardinalidades
- Dependencias de existencia

Correspondencia de cardinalidades

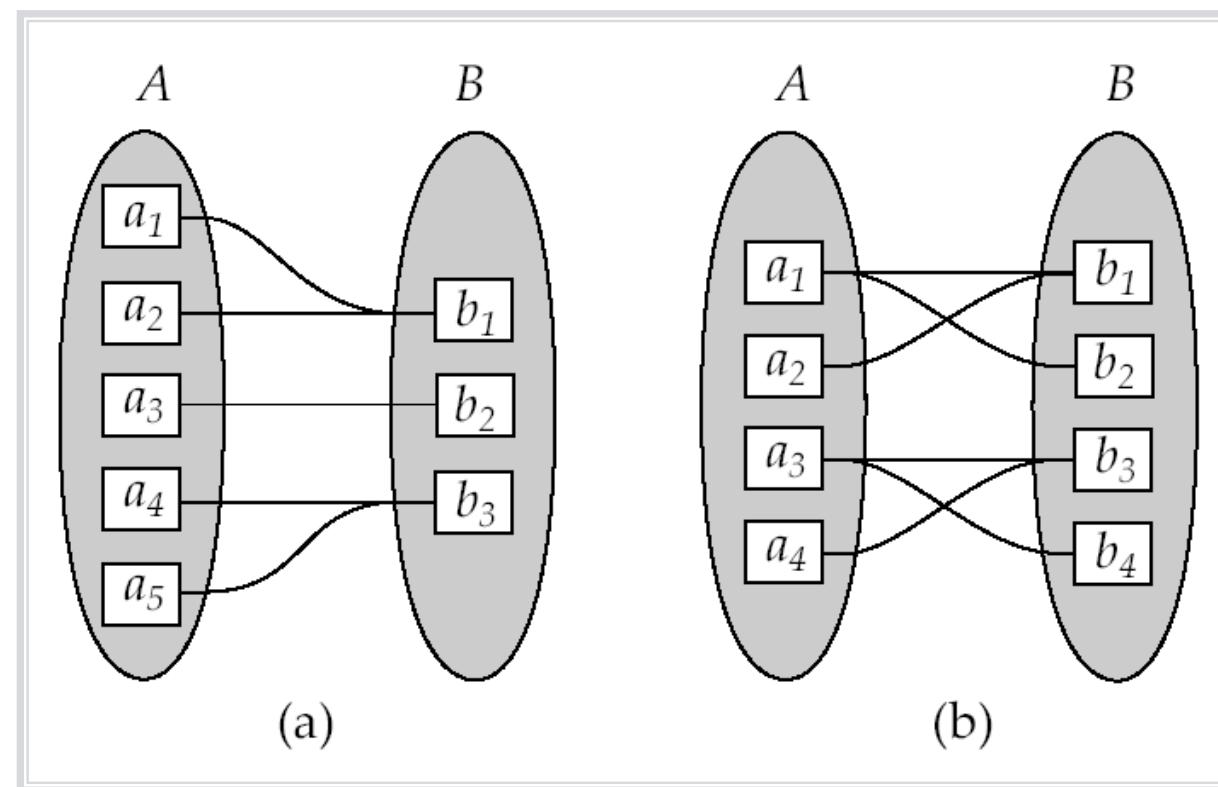
- Expresa el número de entidades a las que otra entidad puede estar asociada vía un conjunto de relaciones (sólo binarias)
- Las correspondencias de cardinalidad de un conjunto de relaciones binarias R entre los conjuntos de entidades A y B, podrán ser
 - Uno a uno**
 - Uno a varios**
 - Varios a uno**
 - Varios a varios**

Correspondencia de cardinalidades



Algunos elementos de A y B puede que no se correspondan con ningún elemento del otro conjunto

Correspondencia de cardinalidades

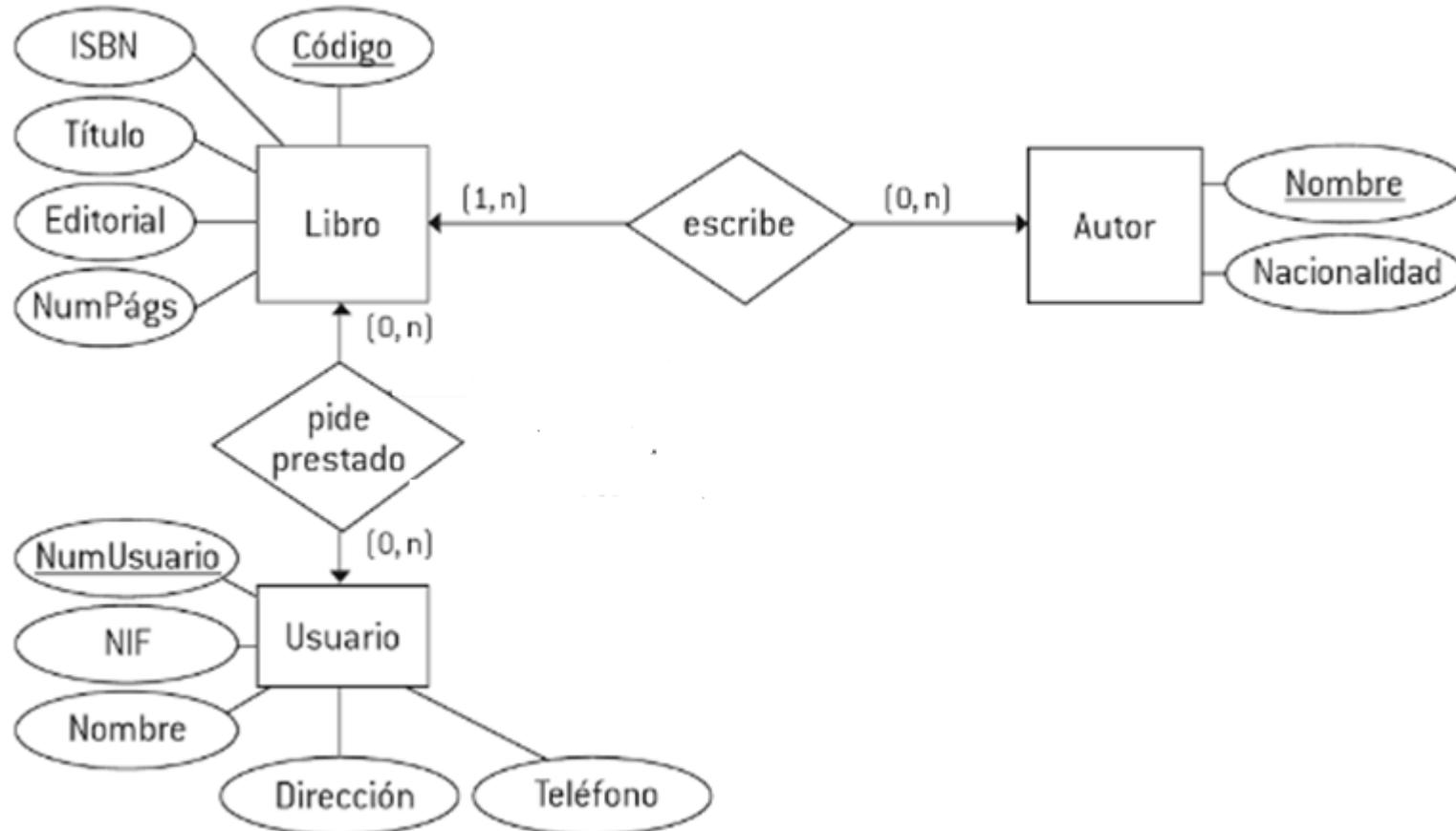


Varios a uno

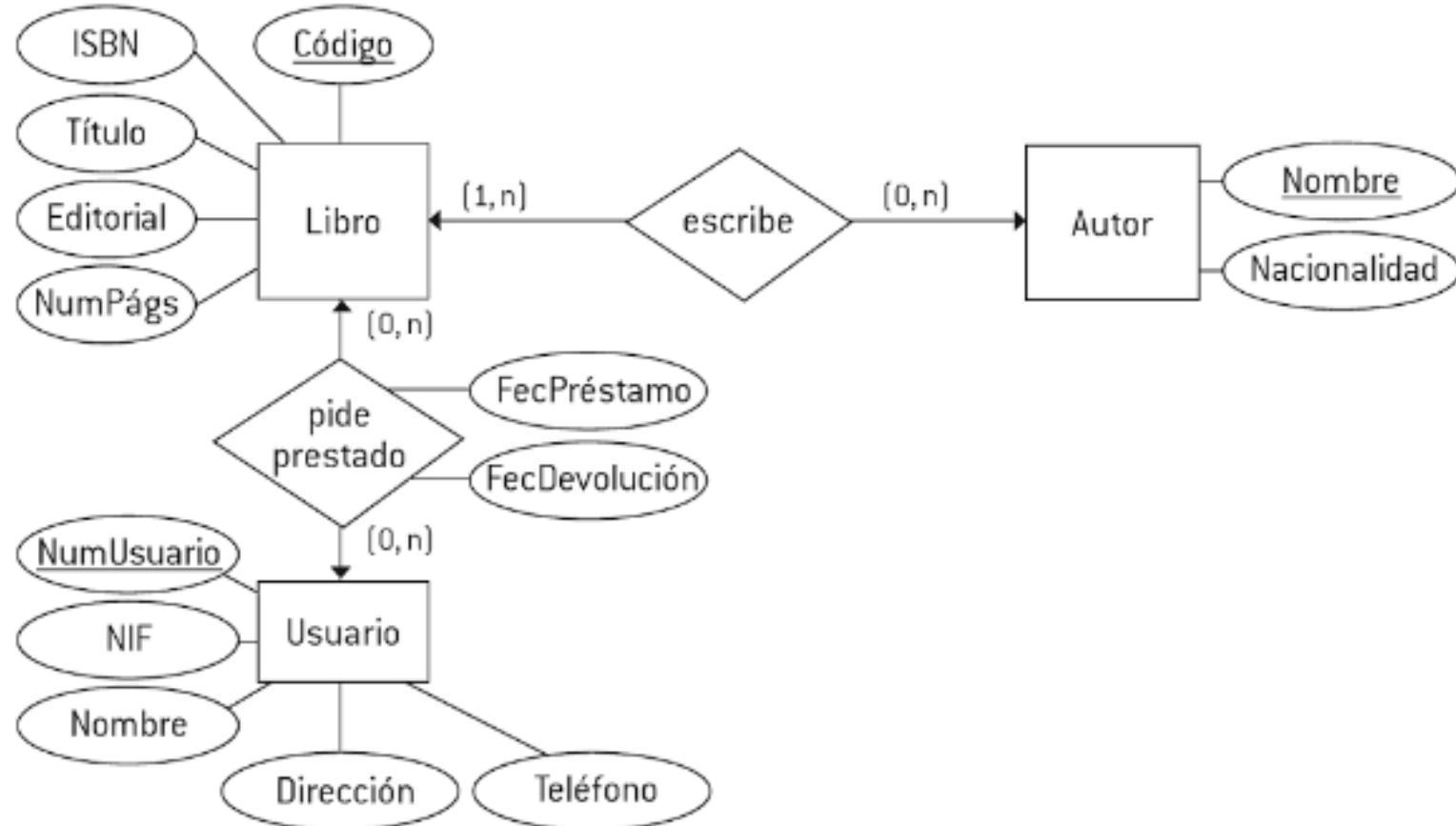
Varios a varios

Algunos elementos de A y B puede que no se correspondan con ningún elemento del otro conjunto

Correspondencia de cardinalidades



Correspondencia de cardinalidades



Claves

Claves

Sirven para **diferenciar las entidades** dentro de un conjunto de entidades y las **relaciones** dentro de un conjunto de relaciones.

Conjunto de entidades

Una clave es una propiedad del conjunto de entidades, más que de las entidades individuales.

Cualesquiera dos entidades en el conjunto **no** pueden tener el **mismo valor** en sus atributos clave al mismo tiempo.

Claves

Superclave

Conjunto de uno o más atributos que tomados colectivamente, permiten **identificar de forma única** una entidad en el conjunto de entidades.

- El DNI de cliente es una superclave.
- El DNI y el nombre es una superclave del conjunto de entidades cliente.

Clave candidata

Es una **superclave mínima**

- Los atributos **calle** y **nombre** distinguen a los miembros del conjunto de entidades cliente.
- Los conjuntos {DNI} y {nombre, calle} son claves candidatas.
- Aunque DNI y nombre puedan distinguir entidades cliente, su combinación no forma una clave candidata, ya que el DNI sólo es una clave candidata.
- El DNI puede ser una clave candidata (salvo que existan no residentes que no disponen de DNI) una alternativa sería generar números de cliente, códigos de identificación, ...

Claves

Clave primaria

Es una **clave candidata** que es elegida por el **diseñador** de la base de datos como elemento principal para identificar las entidades dentro del conjunto de entidades.

Conjuntos de relaciones

Sea R un conjunto de relaciones que implica los conjuntos de entidades E_1, E_2, \dots, E_n .

Sea clave-primaria (E_i) el conjunto de atributos que forma la clave primaria para el conjunto de entidades E_i

Los nombres de los atributos de todas las claves primarias son únicos.

Claves

La composición de una **clave primaria para un conjunto de relaciones** depende de la estructura de los atributos asociados al conjunto de relaciones R

- Si el conjunto de relaciones R no tiene atributos asociados, entonces el conjunto de atributos
$$\text{clave-primaria}(E_1) \cup \text{clave-primaria}(E_2) \cup \dots \cup \text{Clave-primaria}(E_n)$$
describe una relación individual en el conjunto R
- Si el conjunto de R tiene atributos a₁,a₂,...,a_m asociados a él, entonces el conjunto de atributos
$$\text{clave-primaria}(E_1) \cup \text{clave-primaria}(E_2) \cup \dots \cup \text{Clave-primaria}(E_n) \cup \{a_1, a_2, \dots, a_m\}$$
describe una relación individual en el conjunto R

Claves

- En todo caso el conjunto de atributos

clave-primaria(E_1) \cup clave-primaria(E_2) \cup \cup Clave-primaria(E_n)

forman una **superclave** para el conjunto de relaciones

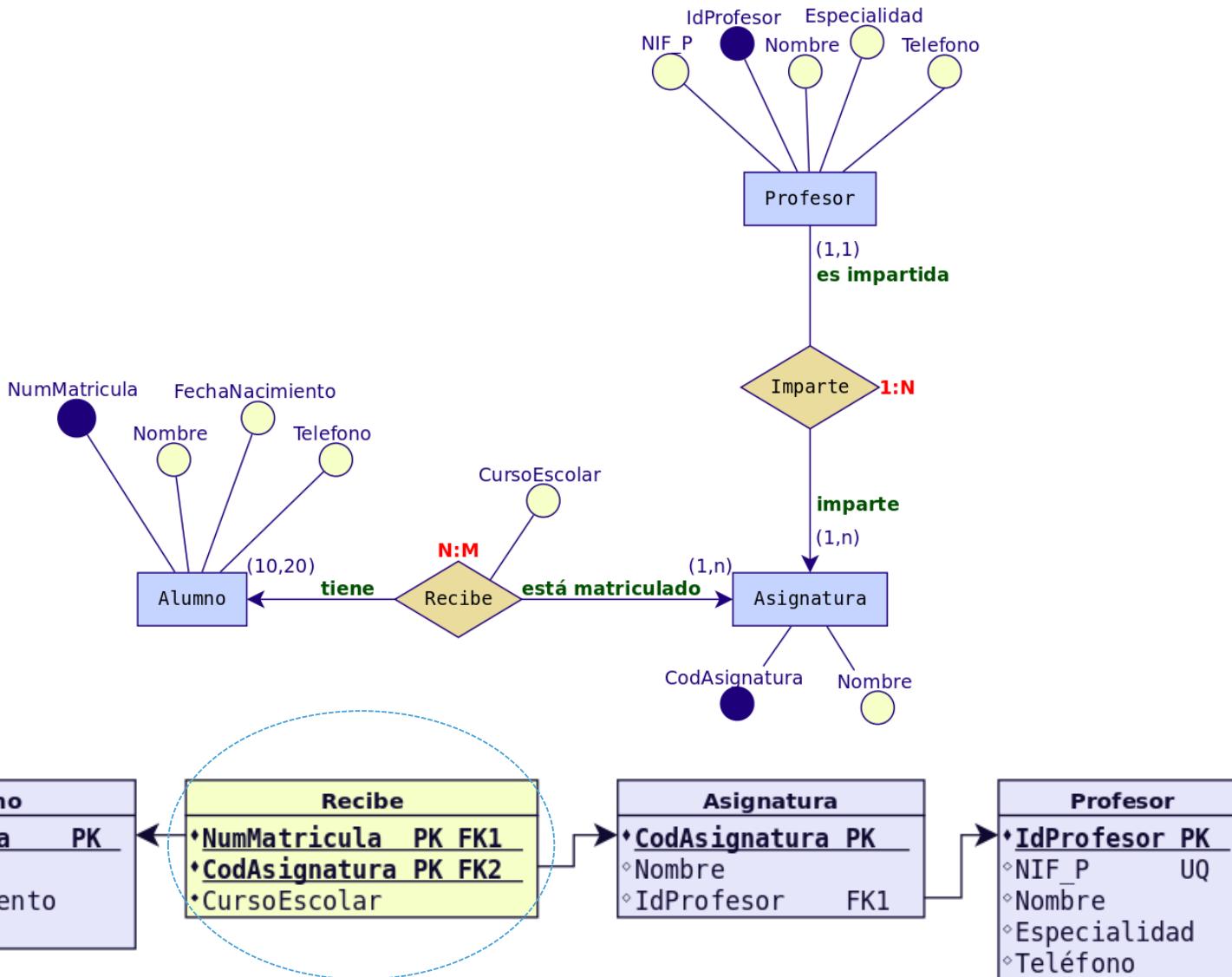
- La estructura de la clave primaria para un conjunto de relaciones depende de la cardinalidad asociada al conjunto de relaciones.

Requisitos: Ejemplo

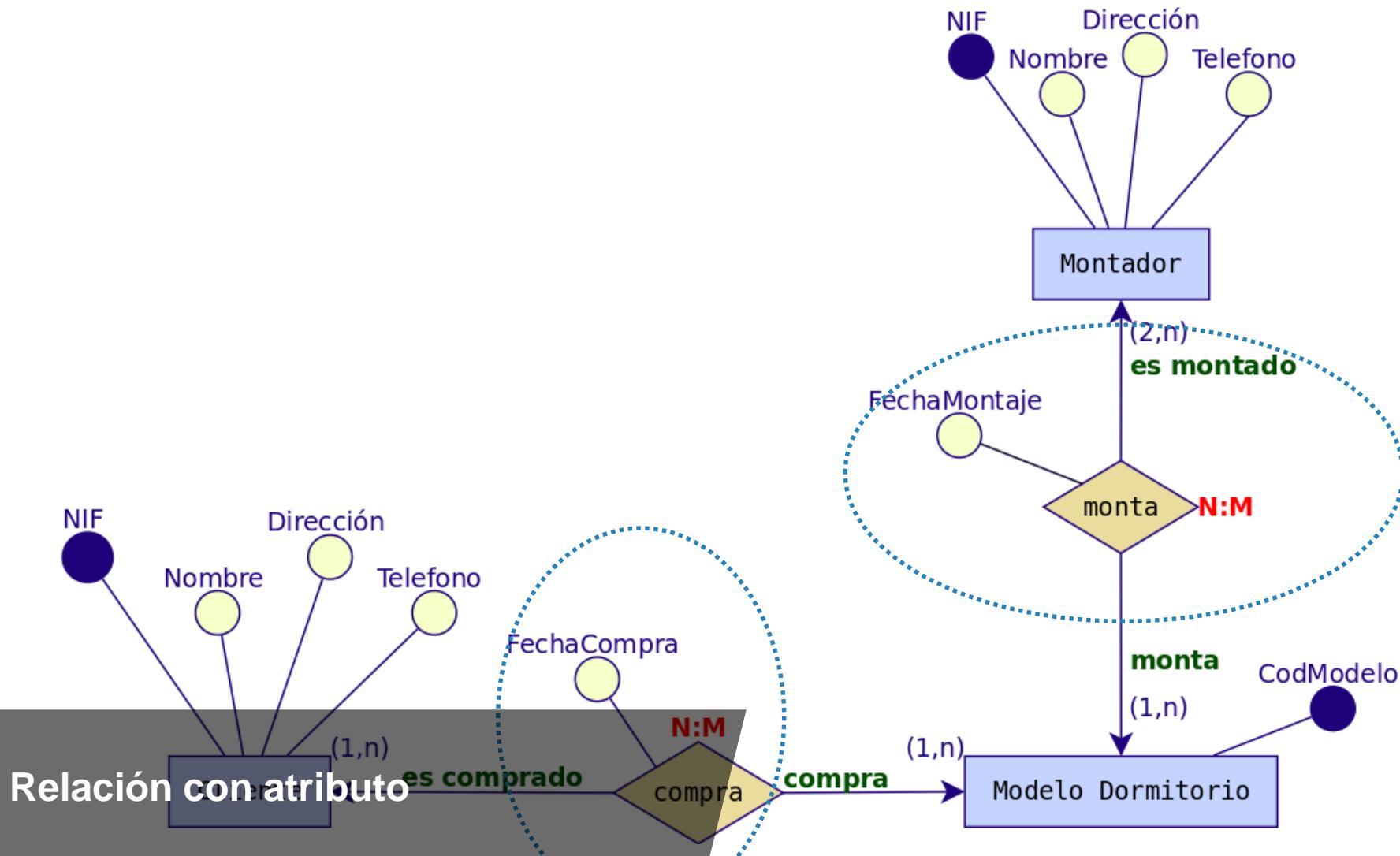
Requisitos necesarios para el diseño e implementación de un sistema relacional que permita consultar los alumnos matriculados en asignaturas así como el profesor que imparte las mismas:

- Un alumno puede estar matriculado en una o varias asignaturas.
- Además puede estar matriculado en la misma asignatura más de un curso escolar (si repite).
- Se quiere saber el curso escolar en el que cada alumno está matriculado de cada asignatura.
- En una asignatura habrá como mínimo 10 y como máximo 20 alumnos.
- Una asignatura es impartida por un único profesor.
- Un profesor podrá impartir varias asignaturas.

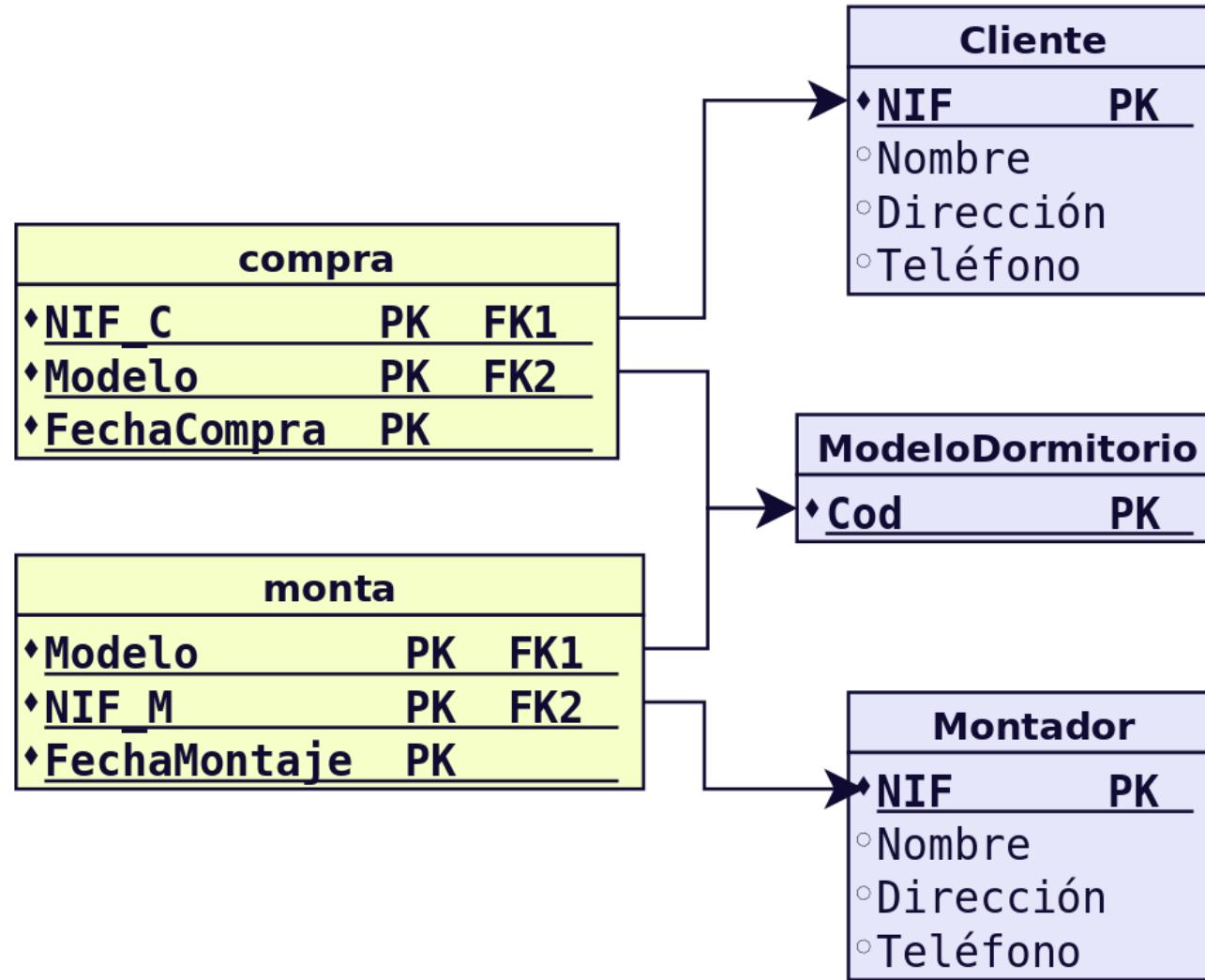
Claves



Un ejemplo

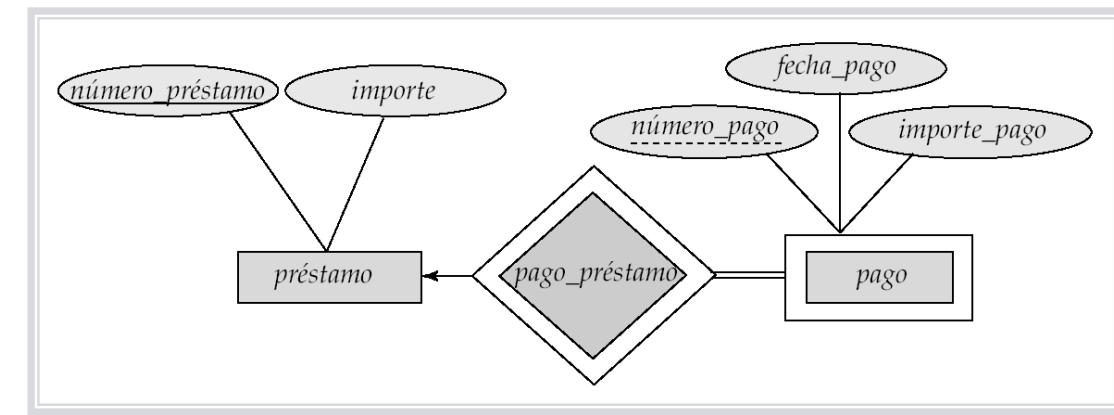
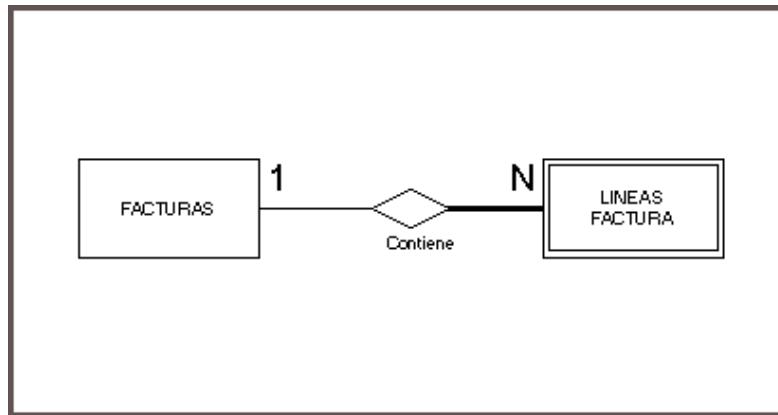


Un ejemplo

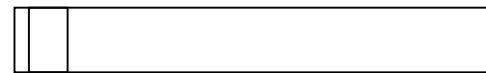


Entidades débiles

- La **clave primaria** de un conjunto de entidades débiles se forma con la clave primaria del conjunto de entidades fuertes del que depende la existencia del conjunto de entidades débiles, más el discriminador de dicho conjunto de entidades débiles.



Un conjunto de entidades débil se indica en los **diagramas E-R** mediante un **rectángulo dibujado con línea doble** y la correspondiente relación de identificación mediante un **rombo dibujado con línea doble**



Entidades fuertes

Entidad fuerte, conjunto de entidades que tienen una clave primaria.

Un miembro de un conjunto de entidades fuerte es una **entidad dominante**.

El conjunto de entidades débil pago es dependiente del conjunto de entidades fuerte préstamo a través del conjunto de relaciones préstamo-pago.

Conclusiones

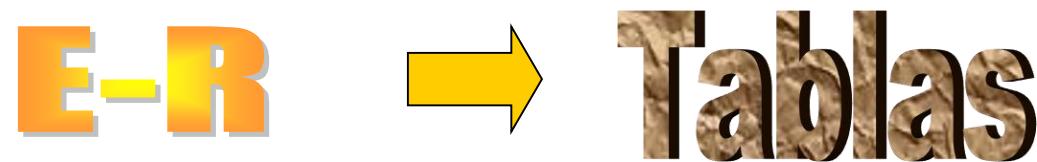
Ventajas del modelo E-R

- Diseño de alto nivel: Expresa con bastante precisión el esquema conceptual
- Los diagramas de E-R permiten mantener una visión global del diseño y favorece la comunicación entre los diseñadores.

Desventajas del modelo E-R:

- Carece de un soporte formal y los SGBD no suelen implementarlo directamente.
- Normalmente hay que transformarlo en un modelo de más bajo nivel.

Reducción de un esquema E-R a tablas



- Los modelos E-R y el modelo relacional son representaciones abstractas y lógicas del desarrollo del mundo real.
- Una **relación** se puede considerar como una **tabla** de valores.
- Representación tabular de los conjuntos de **entidades fuertes**.
- Sea **E** un conjunto de entidades fuertes con los atributos descriptivos a_1, a_2, \dots, a_n .
- Esta entidad se representa mediante una tabla **E** con **n columnas** distintas, cada una de las cuales corresponde a **un atributo**.
- **Cada fila** de la tabla corresponde a **una entidad** del conjunto de entidades.

Índice

- Contexto
- Diagramas Entidad / Relación
- Diagramas Relacionales
- Un repaso por SQL
- Un repaso por los Lenguajes de Programación

ALGUNOS EJEMPLOS...

Ejemplo 1

Realizar el diagrama ER y modelar en MySQL para la base de datos que representa la información siguiente:

ALUMNO (Núm_Matrícula, Nombre, FechaNacimiento, Teléfono)

ASIGNATURA (Código_asignatura, Nombre)

PROFESOR (Id_P, NIF_P, Nombre, Especialidad, Teléfono)

Teniendo en cuenta:

- Un alumno puede estar matriculado de una o varias asignaturas.
- Además puede estar matriculado en la misma asignatura más de un curso escolar (si repite).
- Se quiere saber el curso escolar en el que cada alumno está matriculado de cada asignatura.
- En una asignatura habrá como mínimo 10 y como máximo 20 alumnos.
- Una asignatura es impartida por un único profesor.
- Un profesor podrá impartir varias asignaturas.

Ejemplo 2

Realizar el diagrama ER y modelar en MySQL para la base de datos que representa la información siguiente:

REGIÓN (Nombre_Región)

PROVINCIA (CódigoProvincia, Nombre_provincia)

LOCALIDAD (Código_localidad, Nombre)

EMPLEADO (Id_E, DNI_E, Nombre, Teléfono, Salario)

Se quiere guardar información de la localidad donde ha nacido cada uno de los empleados teniendo en cuenta que:

- Un empleado ha nacido en una sola localidad.
- Cada localidad pertenece a una única provincia.
- Cada provincia pertenece a una única región del país.

Ejemplo 3

Realizar el diagrama ER y modelar en MySQL para la base de datos que representa la información siguiente:

Una empresa dedicada a la instalación de dormitorios juveniles a medida quiere realizar una base de datos donde se reflejen las ventas y montajes, para lo cual se tiene en cuenta:

- Cada modelo de dormitorio lo debe montar, al menos, dos montadores.
- El mismo montador puede montar varios modelos de dormitorios.
- De cada modelo dormitorio nos interesa conocer su código de modelo.
- El mismo montador puede montar el mismo modelo en diferentes fechas. Nos interesa conocer la fecha en la que realiza cada montaje.
- De un montador nos interesa su NIF, nombre, dirección, teléfono de contacto y el número de dormitorios que ha montado de cada modelo.
- Cada modelo de dormitorio puede ser comprado por uno o varios clientes y el mismo cliente podrá comprar uno o varios dormitorios. De un cliente nos interesa su NIF, nombre, dirección, teléfono y fecha de compra de cada modelo.

Ejemplo 4

Realizar el diagrama ER y modelar en MySQL para la base de datos que representa la información siguiente:

Disponemos de la siguiente información de una editorial:

- La editorial tiene varias sucursales, con su domicilio, teléfono y un código de sucursal.
- Cada sucursal tiene varios empleados, de los cuales tendremos sus datos personales, DNI y teléfono. Un empleado trabaja en una única sucursal.
- En cada sucursal se publican varias revistas, de las que almacenaremos su título, número de registro, periodicidad y tipo.
- La editorial tiene periodistas (que no trabajan en las sucursales) que pueden escribir artículos para varias revistas. Almacenaremos los mismos datos que para los empleados, añadiendo su especialidad.
- Para cada revista, almacenaremos información de cada número, que incluirá la fecha, número de páginas y el número de ejemplares vendidos.

Ejemplo 5

Realizar el diagrama ER y modelar en MySQL para la base de datos que representa la información siguiente:

La cadena de Video-Clubs Glob-Gusters ha decidido, para mejorar su servicio, emplear una base de datos para almacenar la información referente a las películas que ofrece en alquiler.

Esta información es la siguiente:

- Una película se caracteriza por su título, nacionalidad, productora y fecha. Puede haber varias películas con el mismo título pero rodadas en fechas distintas.
- En una película pueden participar varios actores (nombre, nacionalidad, sexo) algunos de ellos como actores principales.
- Una película está dirigida por un director (nombre, nacionalidad).
- De cada película se dispone de uno o varios ejemplares diferenciados por un número de ejemplar y caracterizados por su estado de conservación.
- Un ejemplar se puede encontrar alquilado a algún socio (DNI, nombre, dirección, teléfono) . Se desea almacenar la fecha de comienzo del alquiler y la de devolución.
- Un socio tiene que ser avalado por otro socio que responda de él en caso de tener problemas en el alquiler.
- Los atributos discriminantes de las entidades débiles se muestran con un círculo verde oscuro.

Ejemplo 6

Realizar el diagrama ER y modelar en MySQL, una entidad bancaria que contenga información sobre los clientes, las cuentas, las sucursales y las transacciones producidas, con las siguientes restricciones:

- Un cliente debe tener como atributos: DNI, nombre, dirección, localidad, fecha de nacimiento, sexo.
- Una sucursal debe poder identificarse a través de un código, además de disponer los atributos de nombre, dirección y localidad.
- Una transacción viene determinada por un número de transacción (único para cada cuenta), la fecha y la cantidad.
- Un cliente puede tener muchas cuentas.
- Una cuenta puede ser de muchos clientes.
- Una cuenta sólo puede estar en una sucursal.

Ejemplo 7

Realizar el diagrama ER y modelar en MySQL que recoja la organización de un sistema de información en el que se quiere tener los datos sobre municipios, viviendas y personas.

- El municipio debe estar identificado por un identificador único, nombre, código postal y provincia.
- Una vivienda se identifica por el tipo de vía, nombre de la vía y número.
- Cada persona está identificada por su DNI, nombre, sexo y fecha de nacimiento.
- Cada persona sólo puede habitar una vivienda, pero puede ser propietaria de varias.
- También nos interesa la relación de las personas con su cabeza de familia..

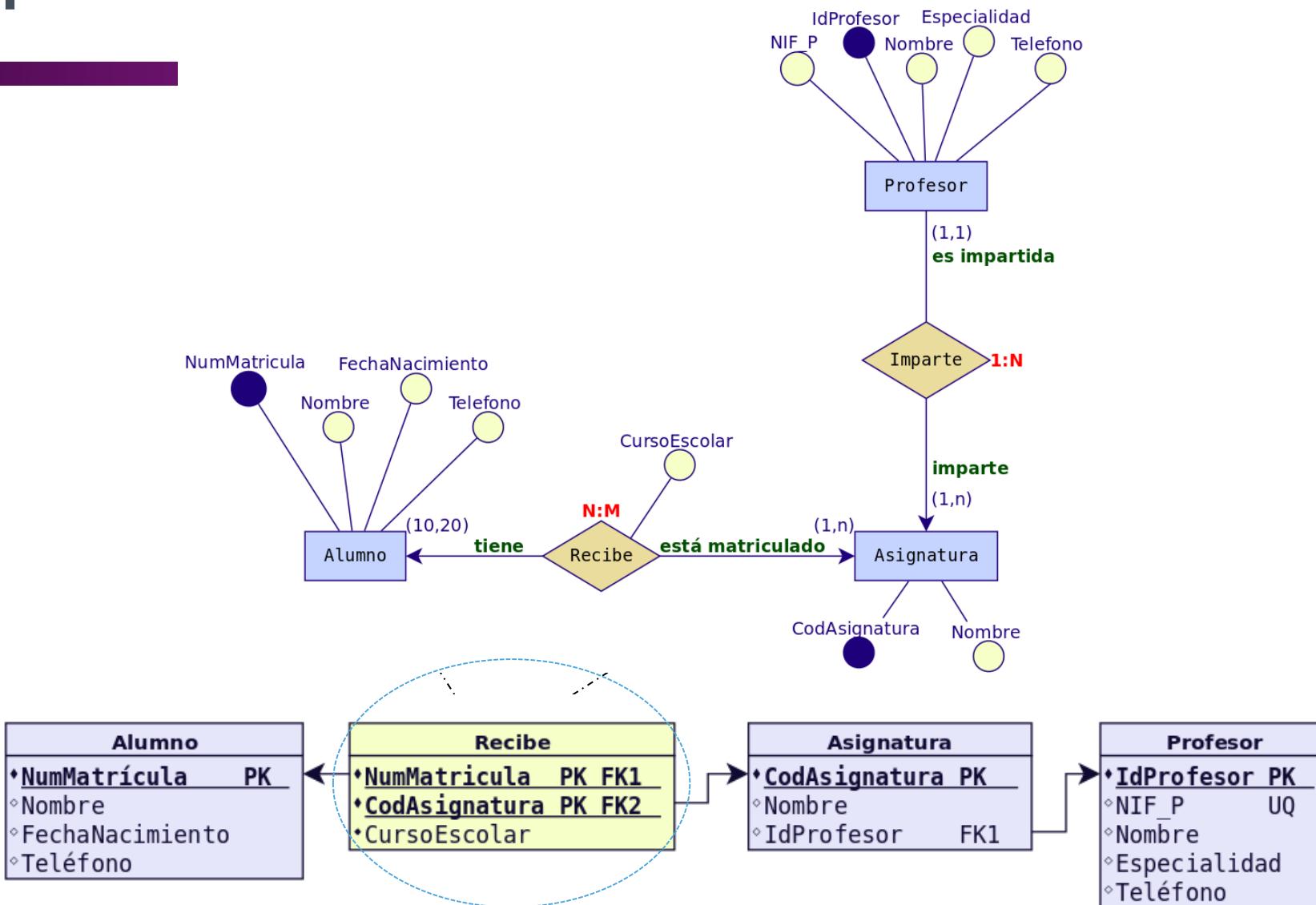
Ejemplo 8

Realizar el diagrama ER y modelar en MySQL que recoja la informatización de la gestión de nóminas dentro de una empresa. Después de un análisis de requisitos con el área de negocio se obtienen los siguientes resultados:

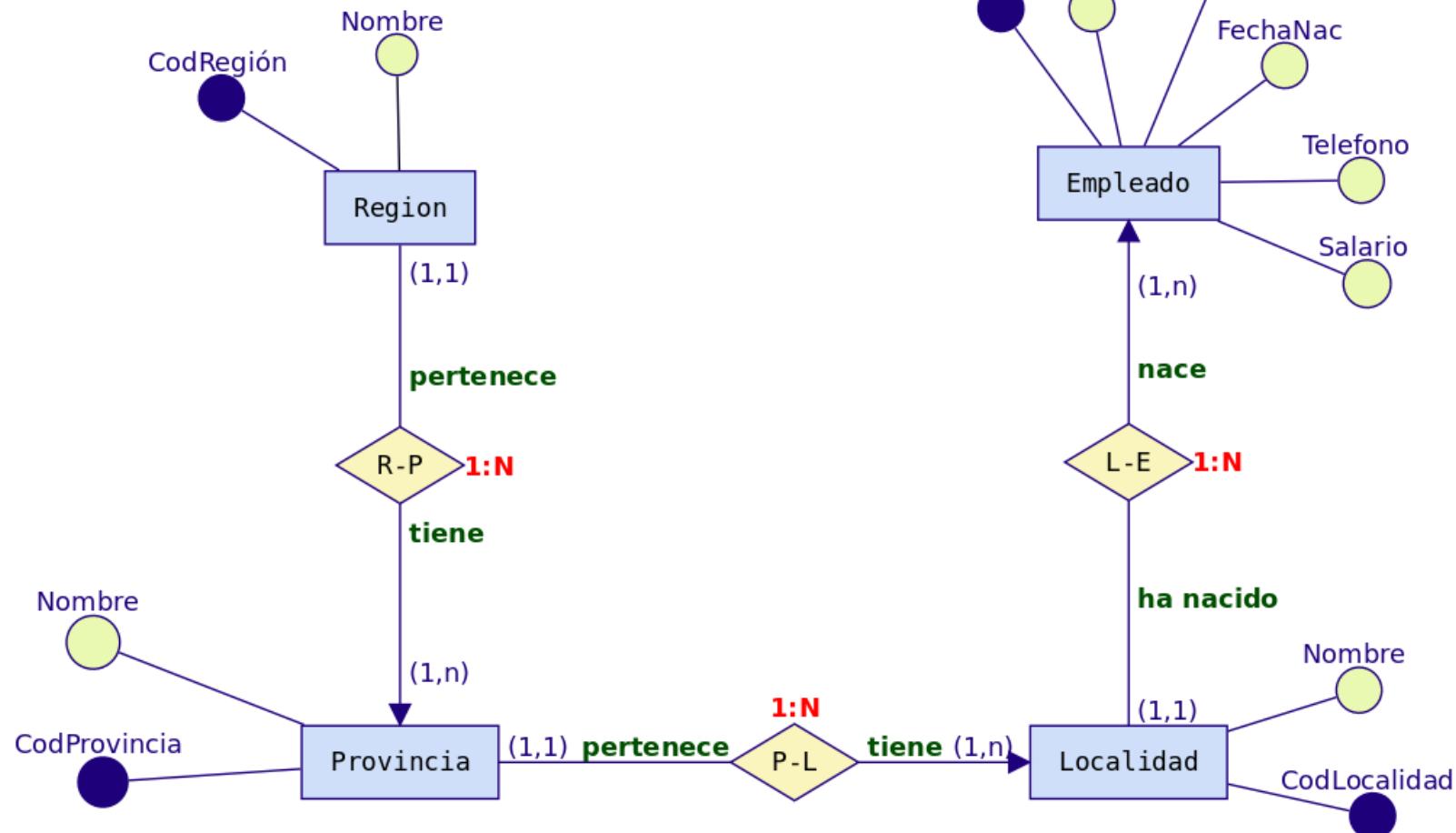
- A cada empleado se le entregan múltiples nóminas a lo largo de su vida laboral en la empresa y al menos una mensualmente.
- A cada empleado se le asigna un número de empleado en el momento de su incorporación a la empresa, y éste es el número usado a efectos internos de identificación. Además, se registran el Número de Identificación Fiscal del empleado, nombre, número de hijos, porcentaje de retención para Hacienda, datos de cuenta corriente en la que se le ingresa el dinero (banco, sucursal y número de cuenta) y departamentos en los que trabaja.
- Un empleado puede trabajar en varios departamentos y en cada uno de ellos trabajará con un función distinta.
- De un departamento se mantiene el nombre y cada una de sus posibles sedes.
- Son datos propios de una nómina el ingreso total percibido por el empleado y el descuento total aplicado.
- La distinción entre dos nóminas se hará, además de mediante el número de identificación del empleado, mediante el ejercicio fiscal y número de mes al que pertenece y con un número de orden en el caso de varias nóminas recibidas el mismo mes.
- Cada nómina consta de varias líneas (al menos una de ingresos) y cada línea se identifica por un número de línea dentro de la correspondiente nómina.
- Una línea puede corresponder a un ingreso o a un descuento. En ambos casos, se recoge la cantidad que corresponde a la línea (en positivo si se trata de un ingreso o en negativo si se trata de un descuento); en el caso de los descuentos, se recoge la base sobre la cual se aplica y el porcentaje que se aplica para el cálculo de éstos.
- Toda línea de ingreso de una nómina responde a un único concepto retributivo.
- En un mismo justificante, puede haber varias líneas que respondan al mismo concepto retributivo.
- De los conceptos retributivos se mantiene un código y una descripción.

SOLUCIONES...

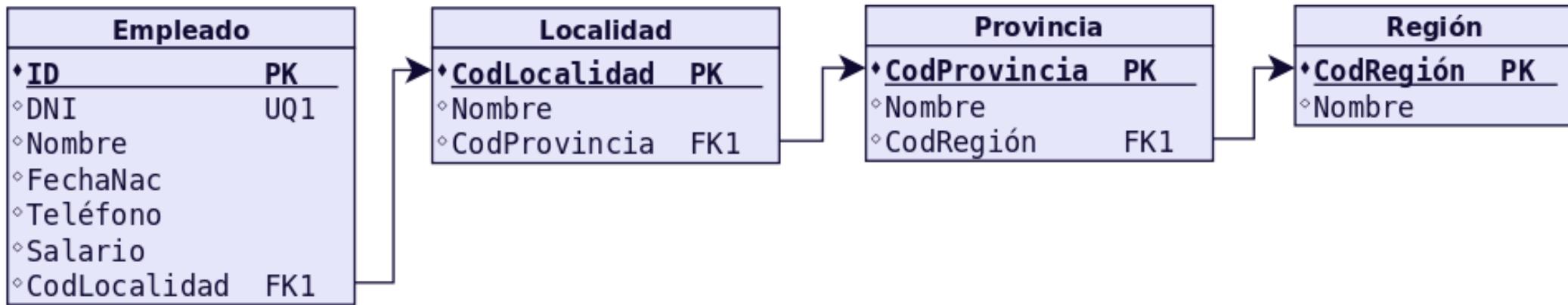
Ejemplo 1. Solución



Ejemplo 2. Solución

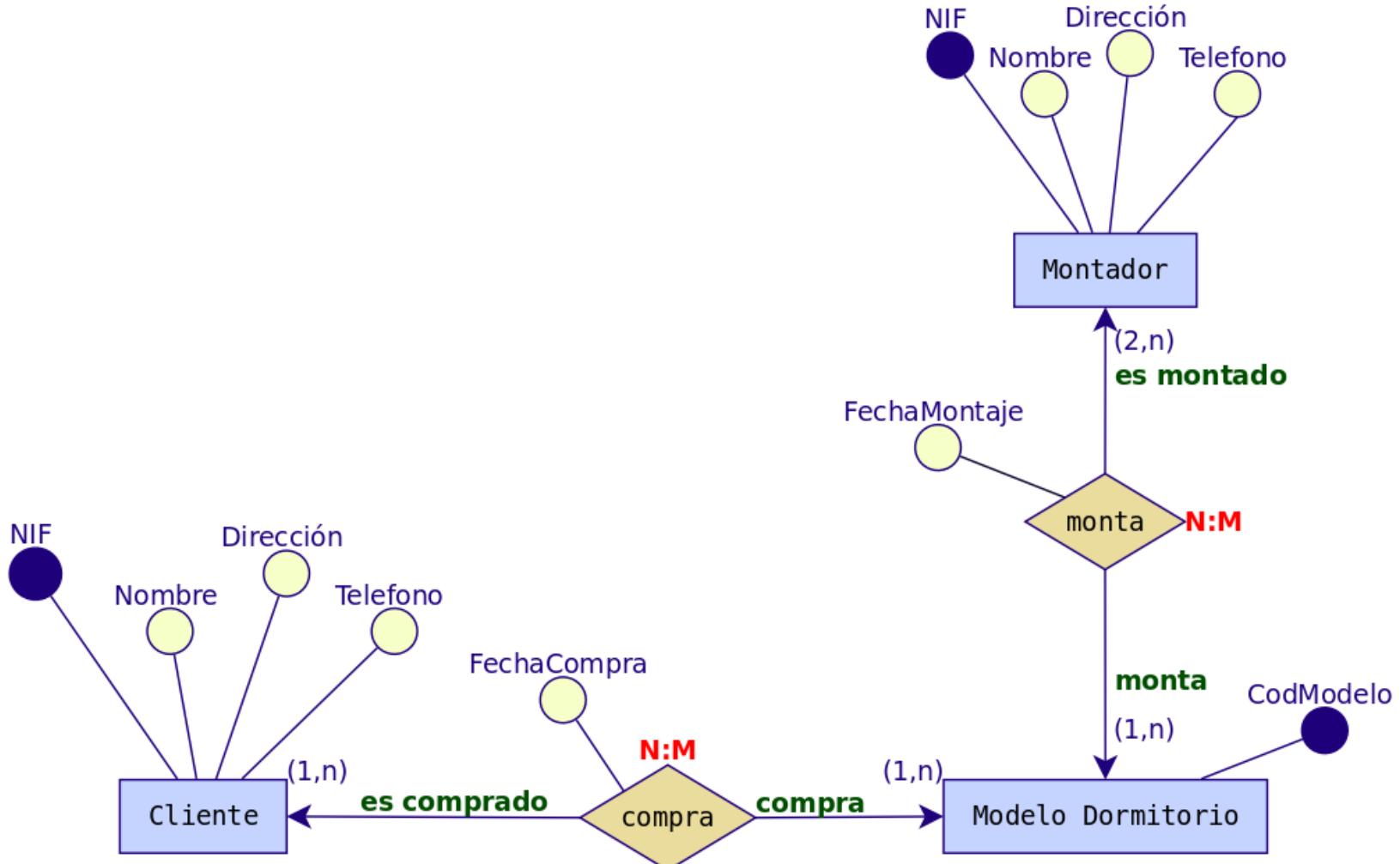


Ejemplo 2. Solución

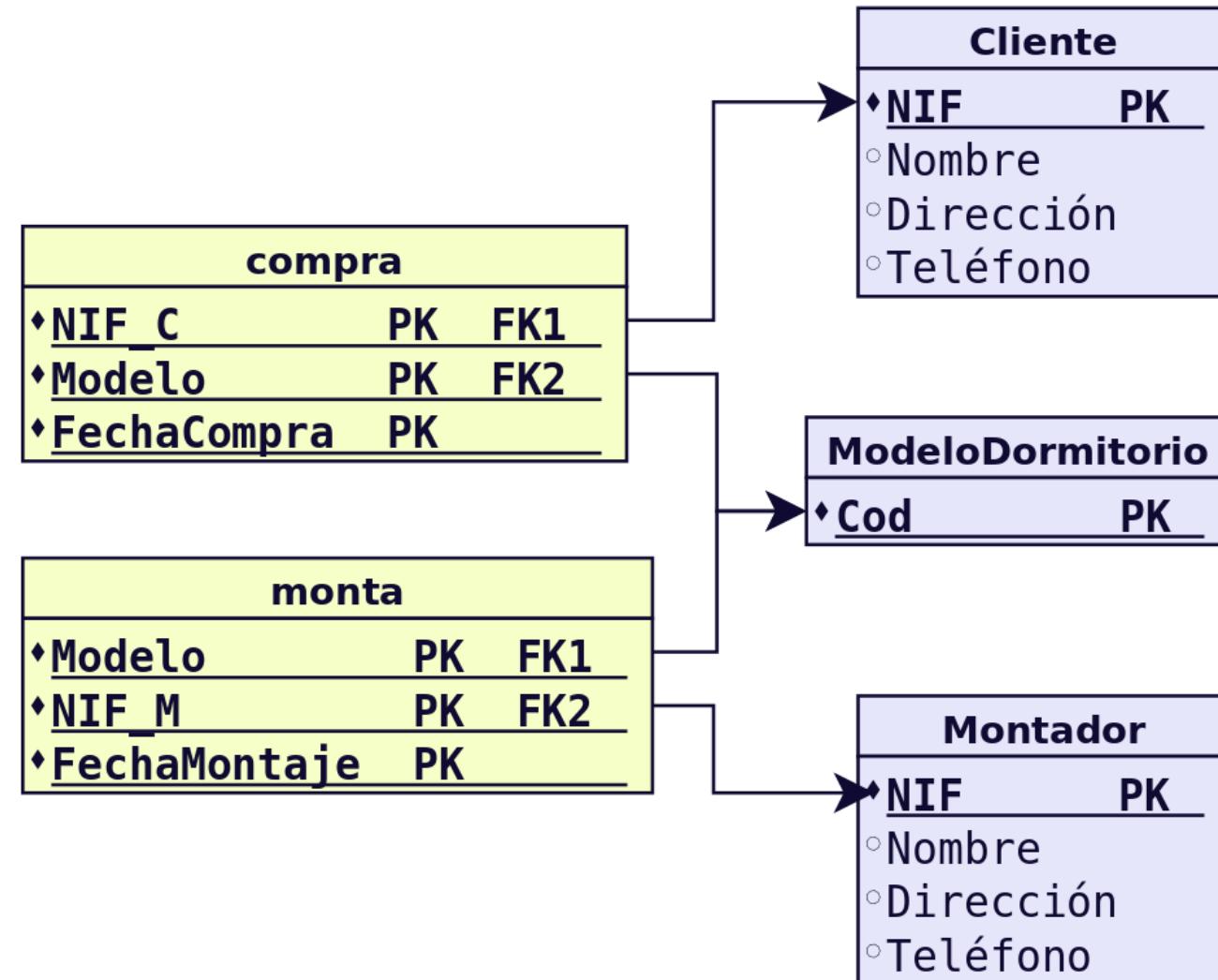


Click me!

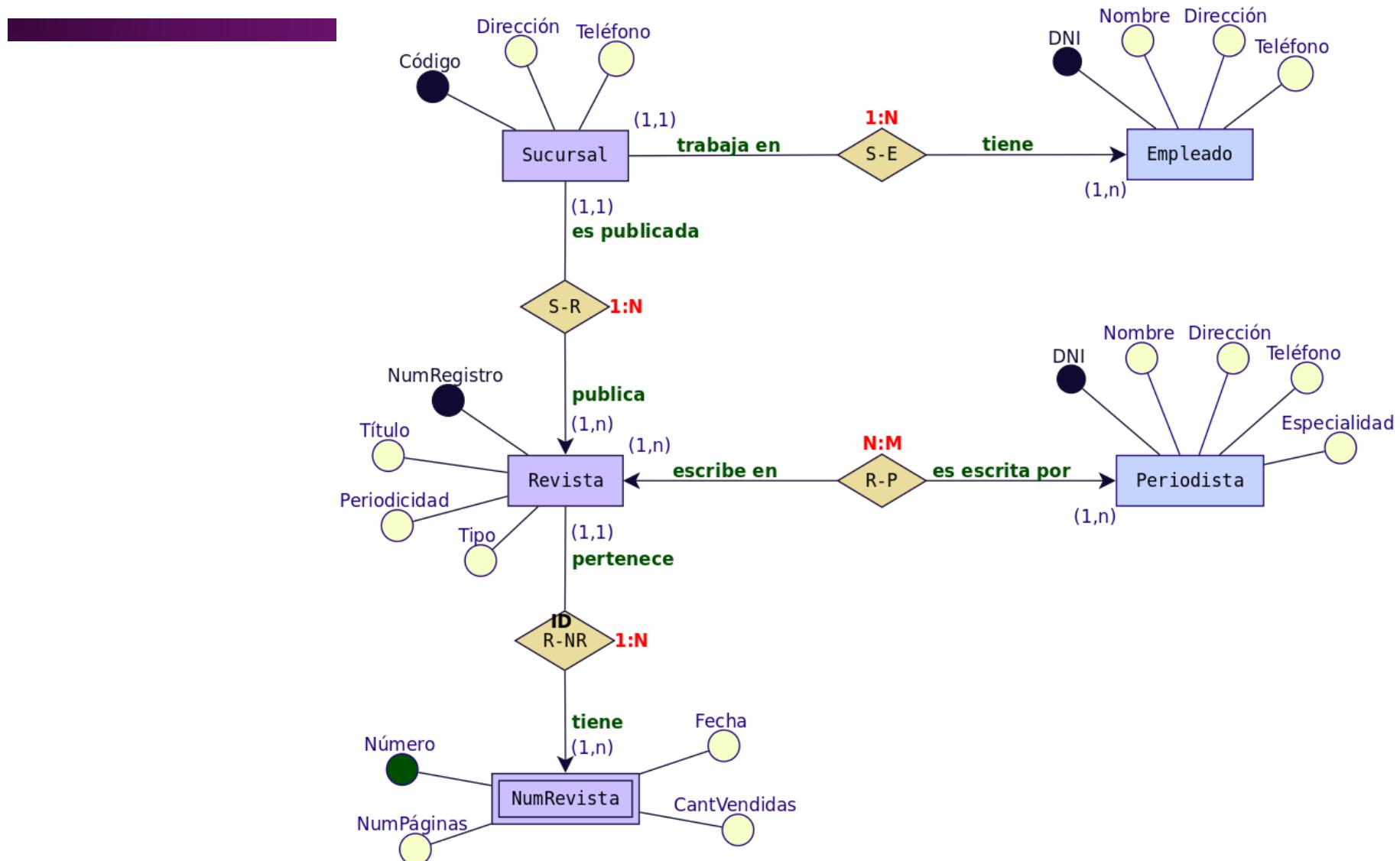
Ejemplo 3. Solución



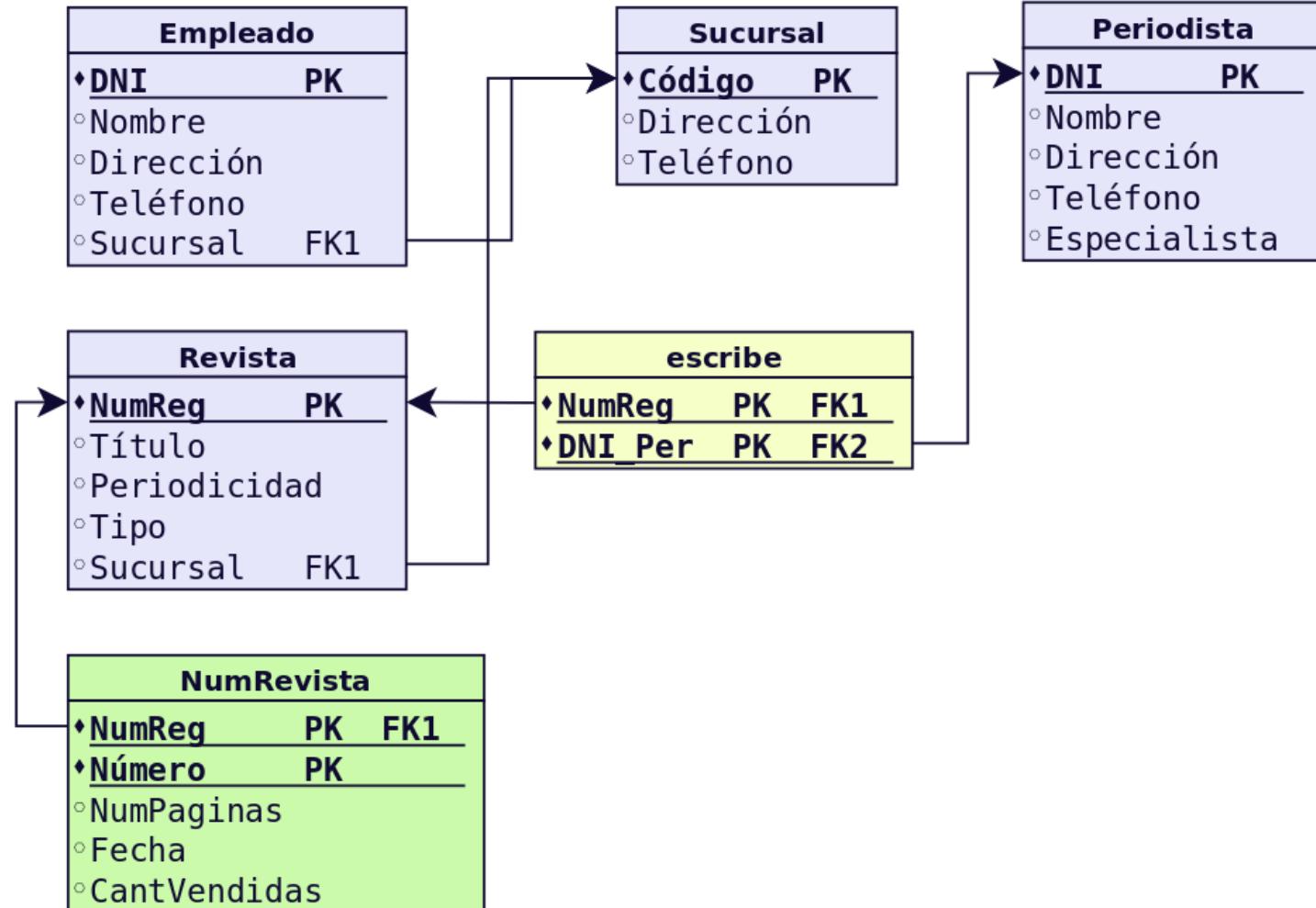
Ejemplo 3. Solución



Ejemplo 4. Solución

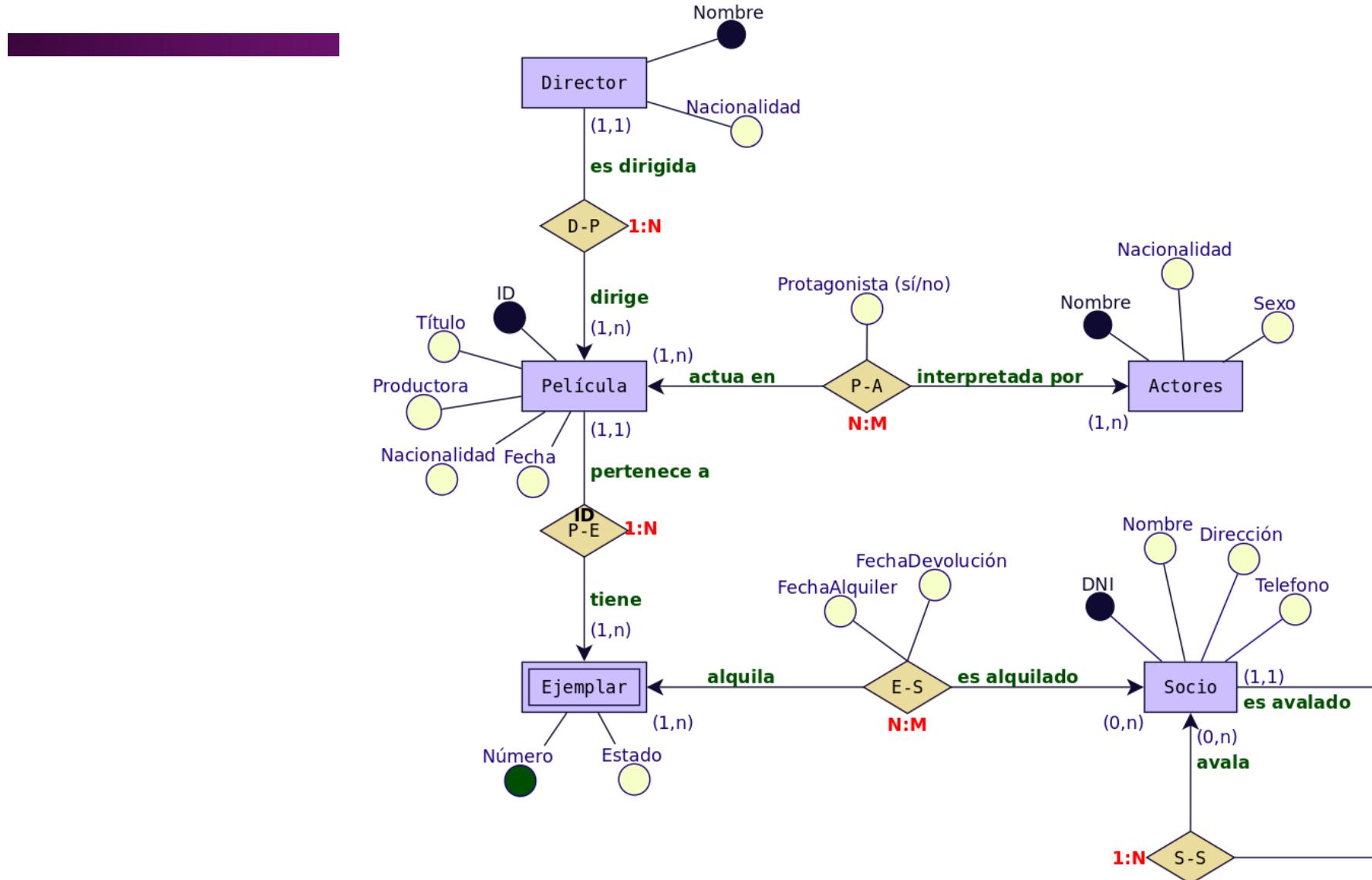


Ejemplo 4. Solución

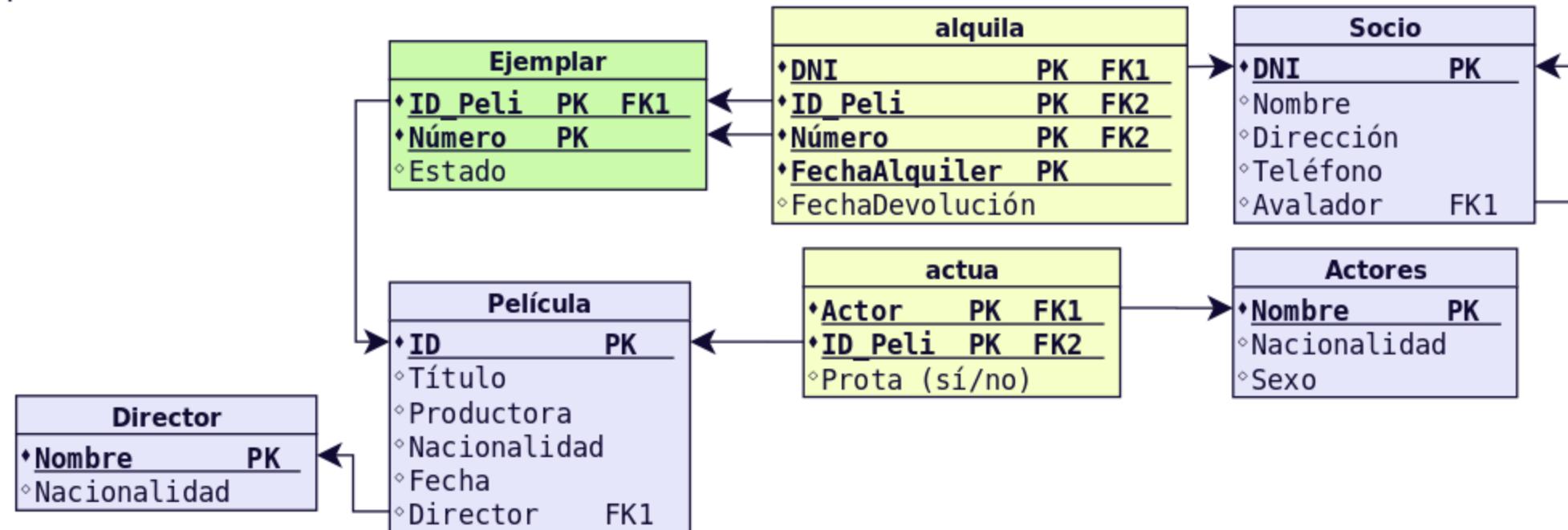


Click me!

Ejemplo 5. Solución

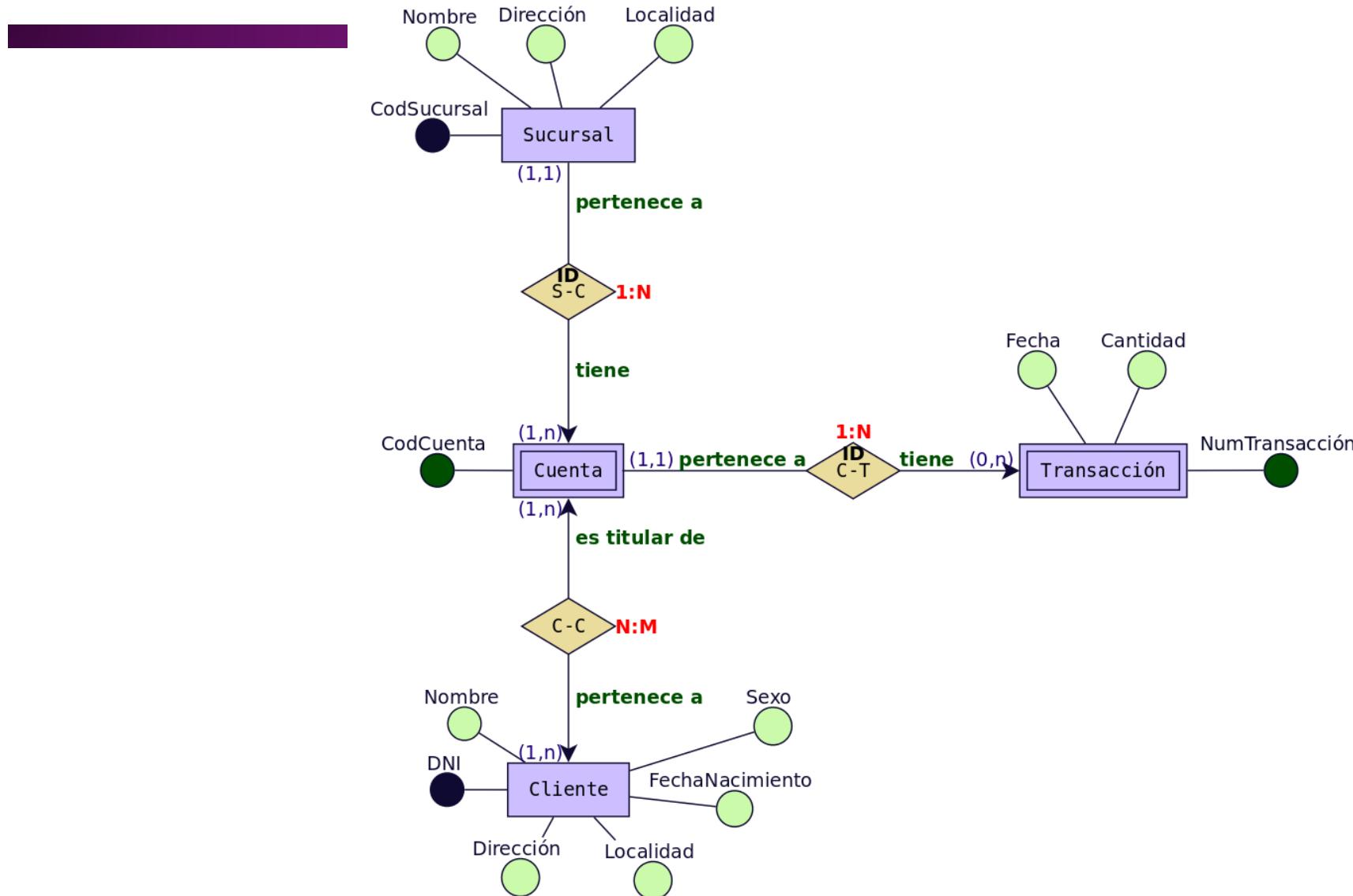


Ejemplo 5. Solución

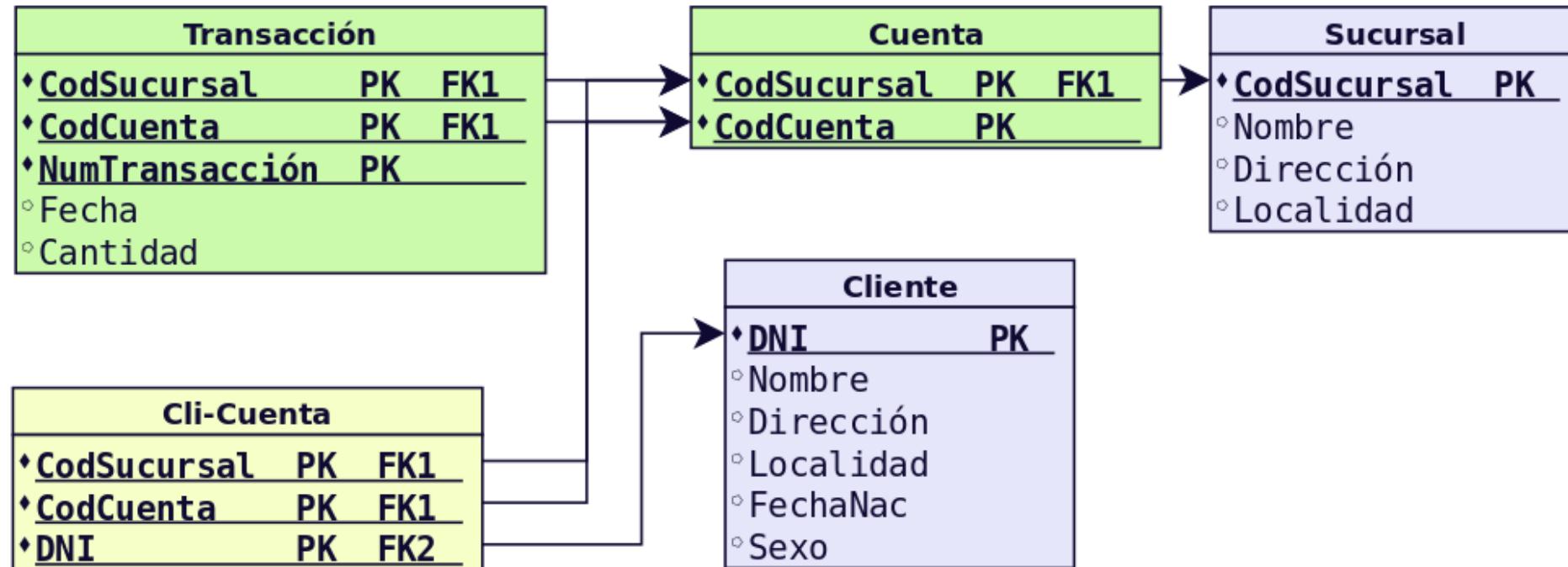


Click me!

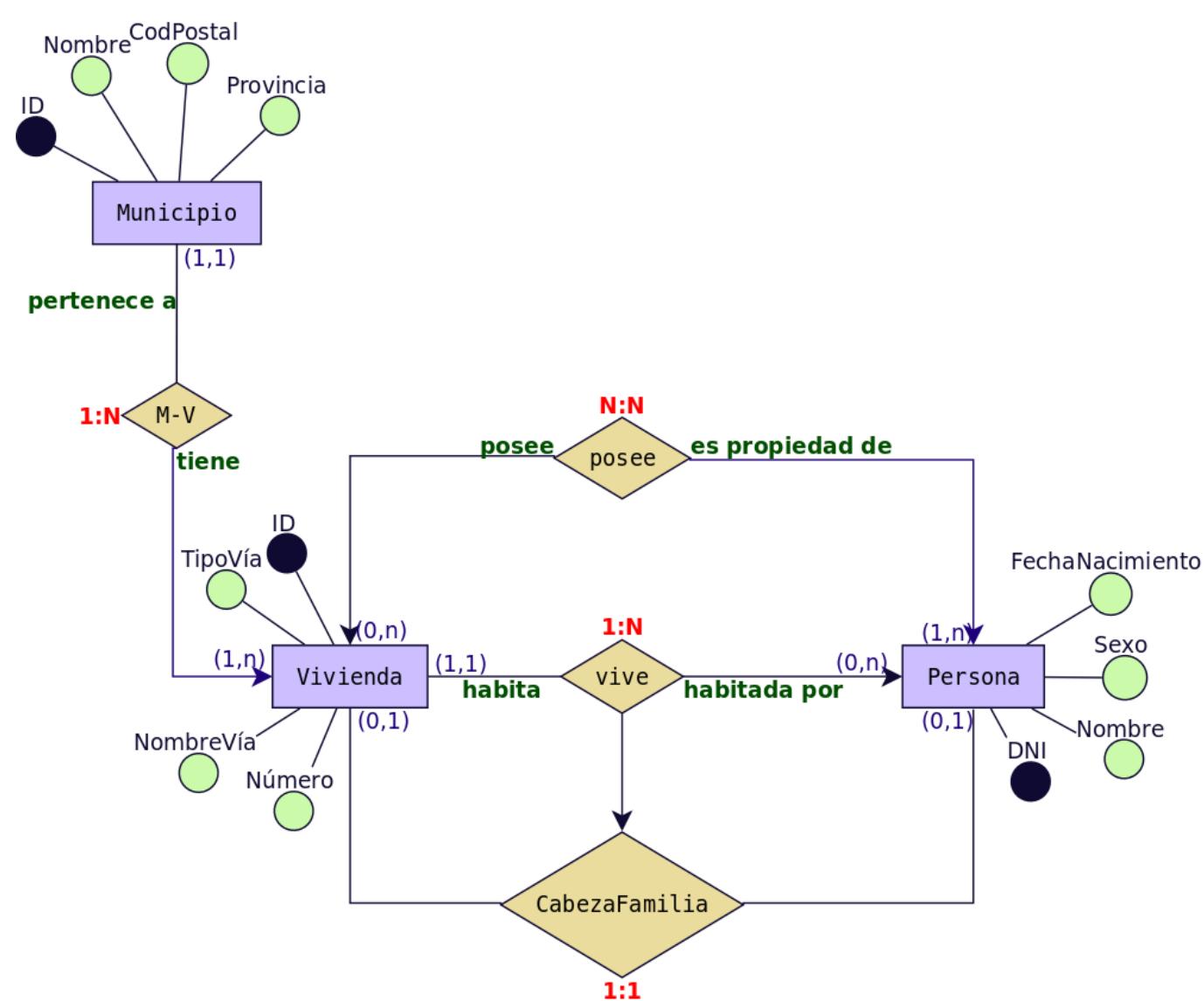
Ejemplo 6. Solución



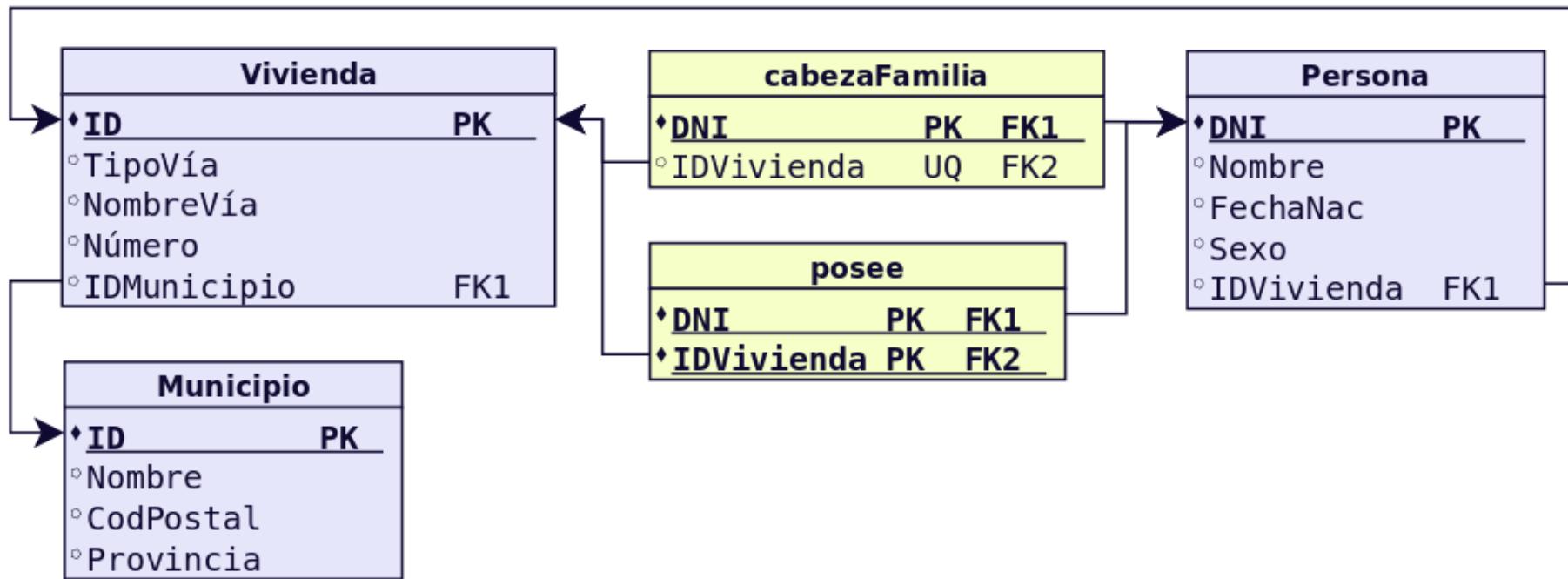
Ejemplo 6. Solución



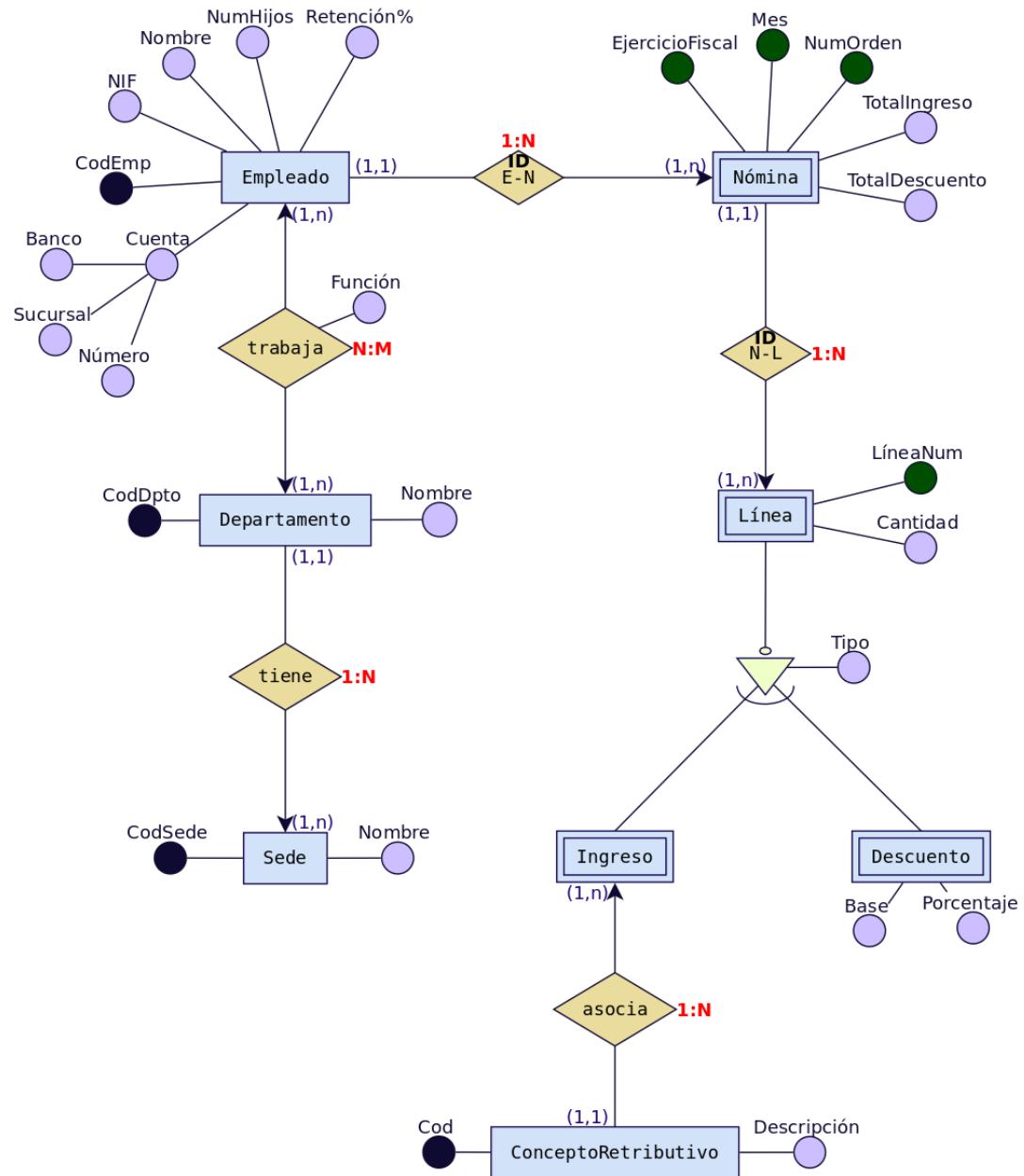
Ejemplo 7. Solución



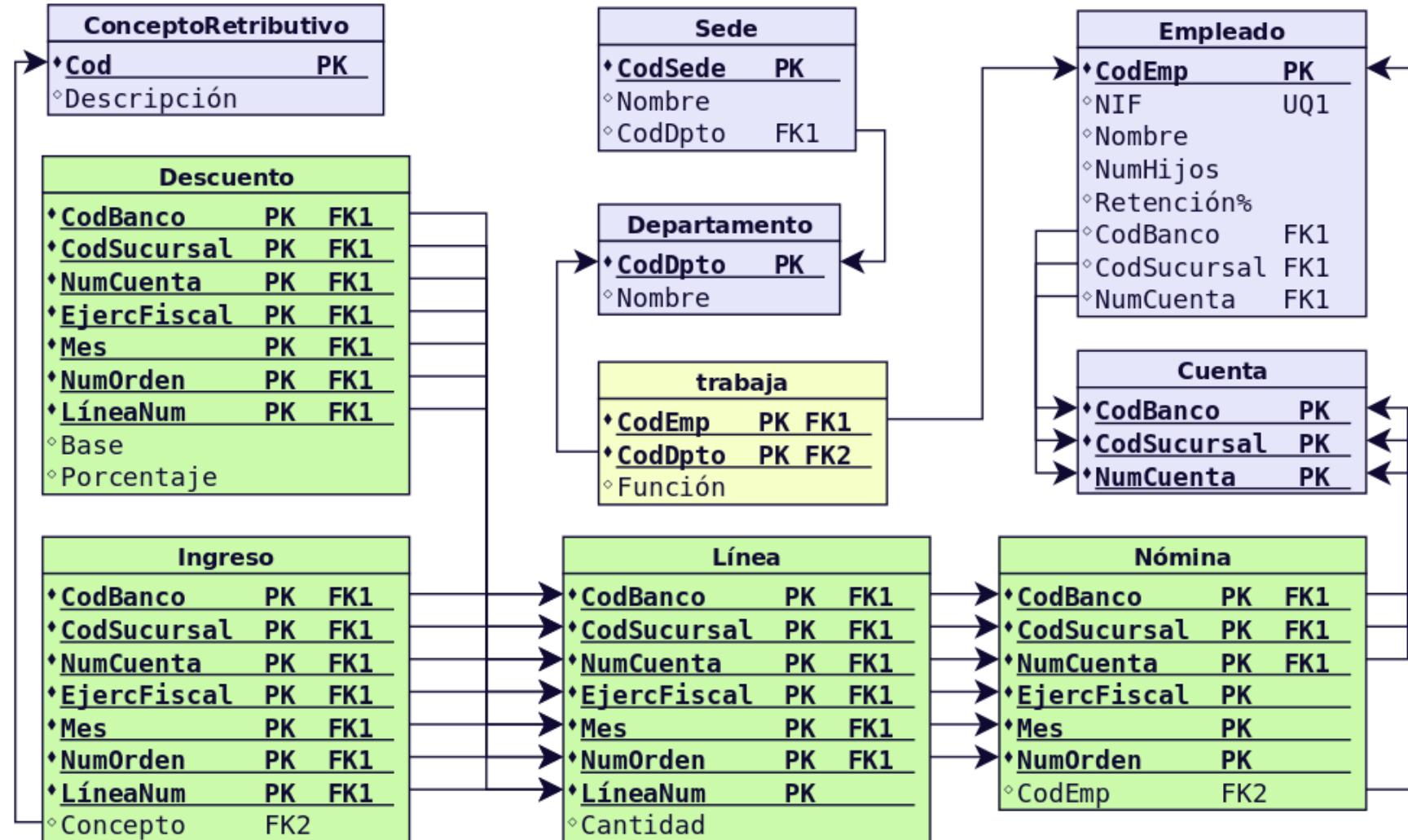
Ejemplo 7. Solución



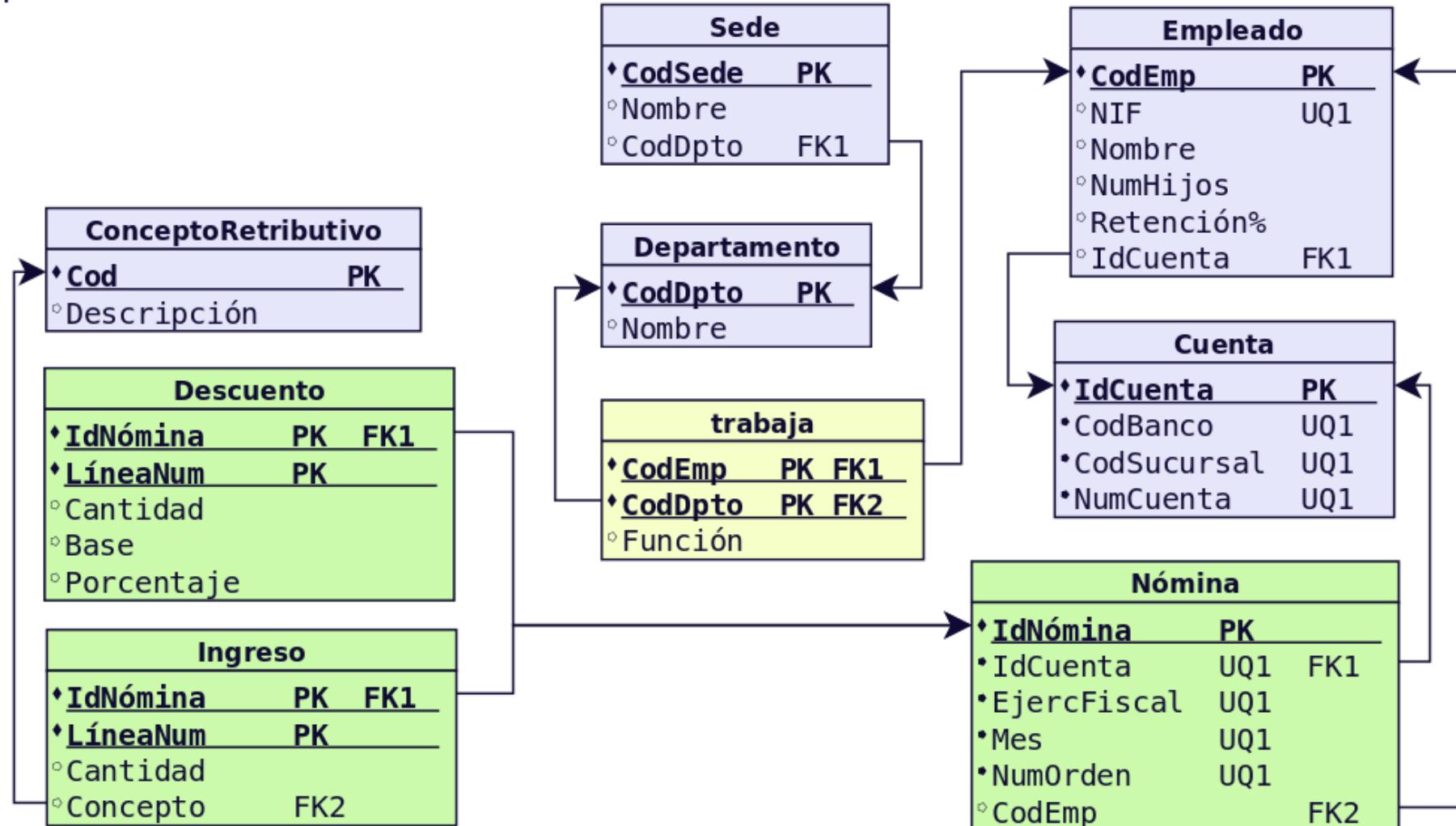
Ejemplo 8. Solución



Ejemplo 8. Solución



Ejemplo 8. Solución simplificada



Click me!

Índice

- Contexto
- Diagramas Entidad / Relación
- Diagramas Relacionales
- Un repaso por SQL
- Un repaso por los Lenguajes de Programación

**YA VISTO EN LOS EJEMPLOS...
RESUMEN DE SENTENCIAS SQL**

Resumen sentencias SQL

Intrucciones SQL	Sintaxis	Intrucciones SQL	Sintaxis
AND / OR	SELECT columna(s) FROM tabla WHERE condición AND OR condición	DELETE	DELETE FROM tabla WHERE columna=valorDELETE FROM tabla (Nota: Borra todos los datos de la tabla)DELETE * FROM tabla (Note:)Borra todos los datos de la tablaDELETE * FROM tabla WHERE condicion
ALTER TABLE	ALTER TABLE tabla ADD columna tipo_datoorALTER TABLE tabla DROP COLUMN columna	DROP DATABASE	DROP DATABASE nombre_bbdd
AS (alias)	SELECT columna AS columna_alias FROM tableaorSELECT columna FROM tablea AS tabla_alias	DROP INDEX	DROP INDEX tabla.indice (SQL Server) DROP INDEX indice ON tabla (MS Access) DROP INDEX indice (DB2/Oracle) ALTER TABLE tabla DROP INDEX indice (MySQL)
BETWEEN	SELECT columna(s) FROM tabla WHERE columna BETWEEN valor1 AND valor2	DROP TABLE	DROP TABLE tabla
CREATE DATABASE	CREATE DATABASE nombre_bbdd	IN	SELECT columna(s) FROM tabla WHERE columna IN (valor1,valor2,...)
CREATE TABLE	CREATE TABLE tabla (columna_1 tipo_dato, columna_2 tipo_dato, columna_3 tipo_dato, ...)	INSERT INTO	INSERT INTO tabla VALUES (valor1, valor2, valor3,...)orINSERT INTO tablea (columna1, columna2, columna3,...) VALUES (valor1, valor2, valor3,...)
CREATE INDEX	CREATE INDEX indice ON tabla (columna)orCREATE UNIQUE INDEX indice ON tabla (columna)	INNER JOIN	SELECT columna(s) FROM tabla1 INNER JOIN tabla2 ON tabla1.columna=tabla2.columna

Resumen sentencias SQL

Instrucciones SQL	Sintaxis	Instrucciones SQL	Sintaxis
LEFT JOIN	<pre>SELECT columna(s) FROM tabla1 LEFT JOIN tabla2 ON tabla1.columna=tabla2.columna</pre>	SELECT *	<pre>SELECT * FROM tabla</pre>
LIMIT	<pre>SELECT columna(s) FROM tabla LIMIT valor</pre>	SELECT DISTINCT	<pre>SELECT DISTINCT columna(s) FROM tabla</pre>
RIGHT JOIN	<pre>SELECT columna(s) FROM tabla1 RIGHT JOIN tabla2 ON tabla1.columna=tabla2.columna</pre>	SELECT INTO	<pre>SELECT * INTO tabla_nueva [IN bbdd_externa] FROM tabla_antigua SELECT columna(s) INTO tabla_nueva [IN bbdd_externa] FROM tabla_antigua</pre>
FULL JOIN	<pre>SELECT columna(s) FROM tabla1 FULL JOIN tabla2 ON tabla1.columna=tabla2.columna</pre>	TRUNCATE TABLE	<pre>TRUNCATE TABLE tabla</pre>
LIKE	<pre>SELECT columna(s) FROM tabla WHERE columna LIKE patrón</pre>	UNION	<pre>SELECT columna(s) FROM tabla1 UNION SELECT columna(s) FROM tabla2</pre>
ORDER BY	<pre>SELECT columna FROM tabla ORDER BY columna [ASC DESC]</pre>	UNION ALL	<pre>SELECT columna(s) FROM tabla1 UNION ALL SELECT columna(s) FROM tabla2</pre>
SELECT	<pre>SELECT columna(s) FROM tabla</pre>	UPDATE	<pre>UPDATE tabla SET columna1=valor, columna2=valor, ... WHERE columna=valor</pre>
		WHERE	<pre>SELECT columna(s) FROM tabla WHERE columna operador valor</pre>

LENGUAJES DE PROGRAMACIÓN

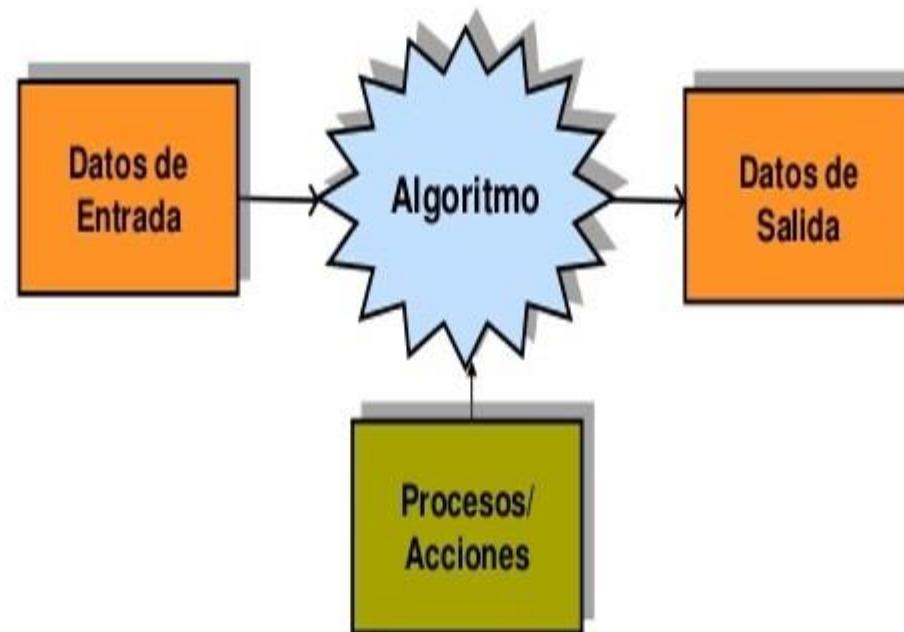
Programación

Ante un problema, la **programación** propone una **solución** mediante un proceso que toma un **algoritmo** y lo codifica en un **lenguaje de programación**, el cual podrá ser ejecutado por un ordenador.

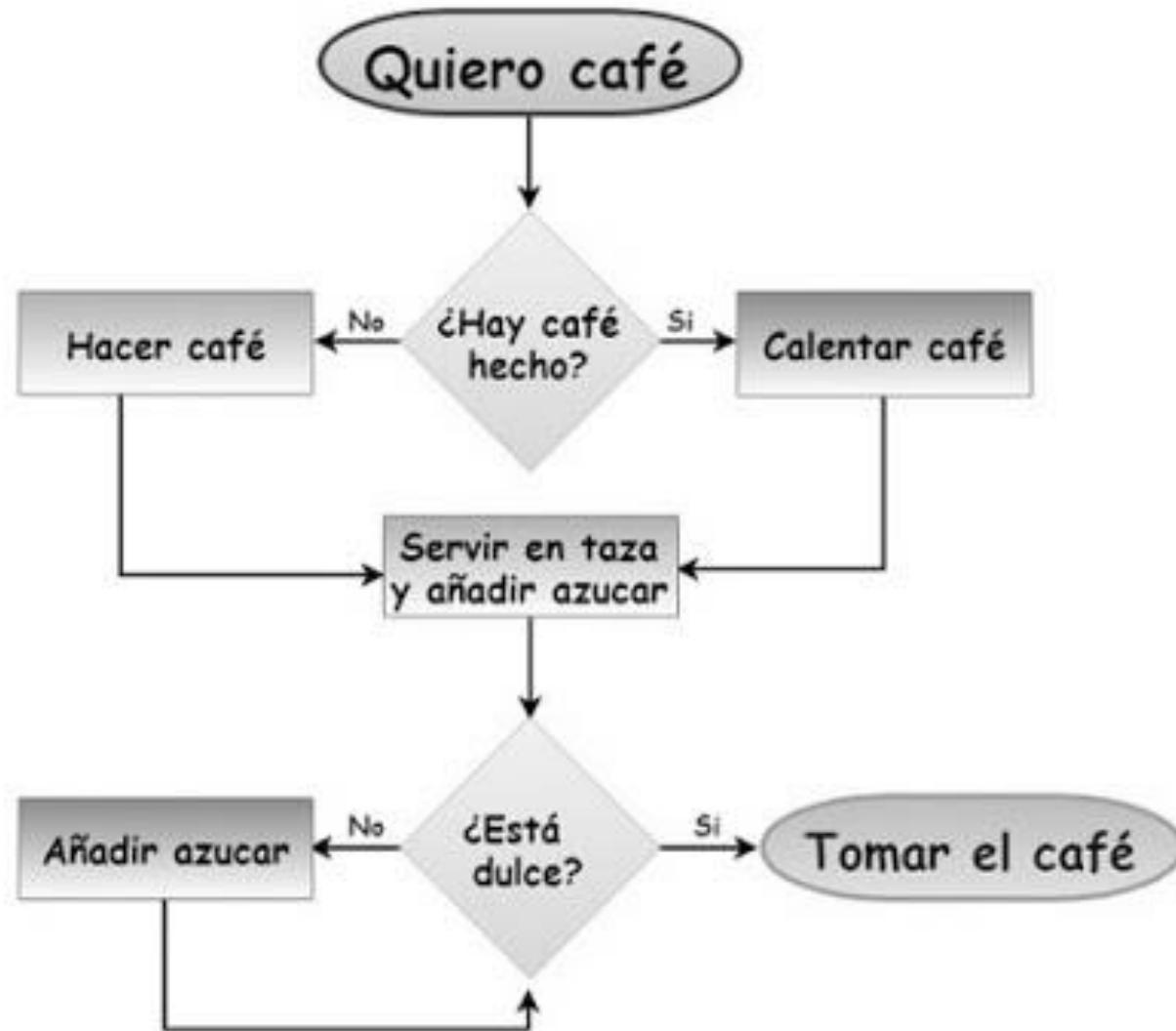


Algoritmo

Los algoritmos describen la solución a un problema, normalmente se utiliza para ello el pseudocódigo de forma que cada programador pueda traducir dicho algoritmo al lenguaje de programación que considere oportuno.



Algoritmo

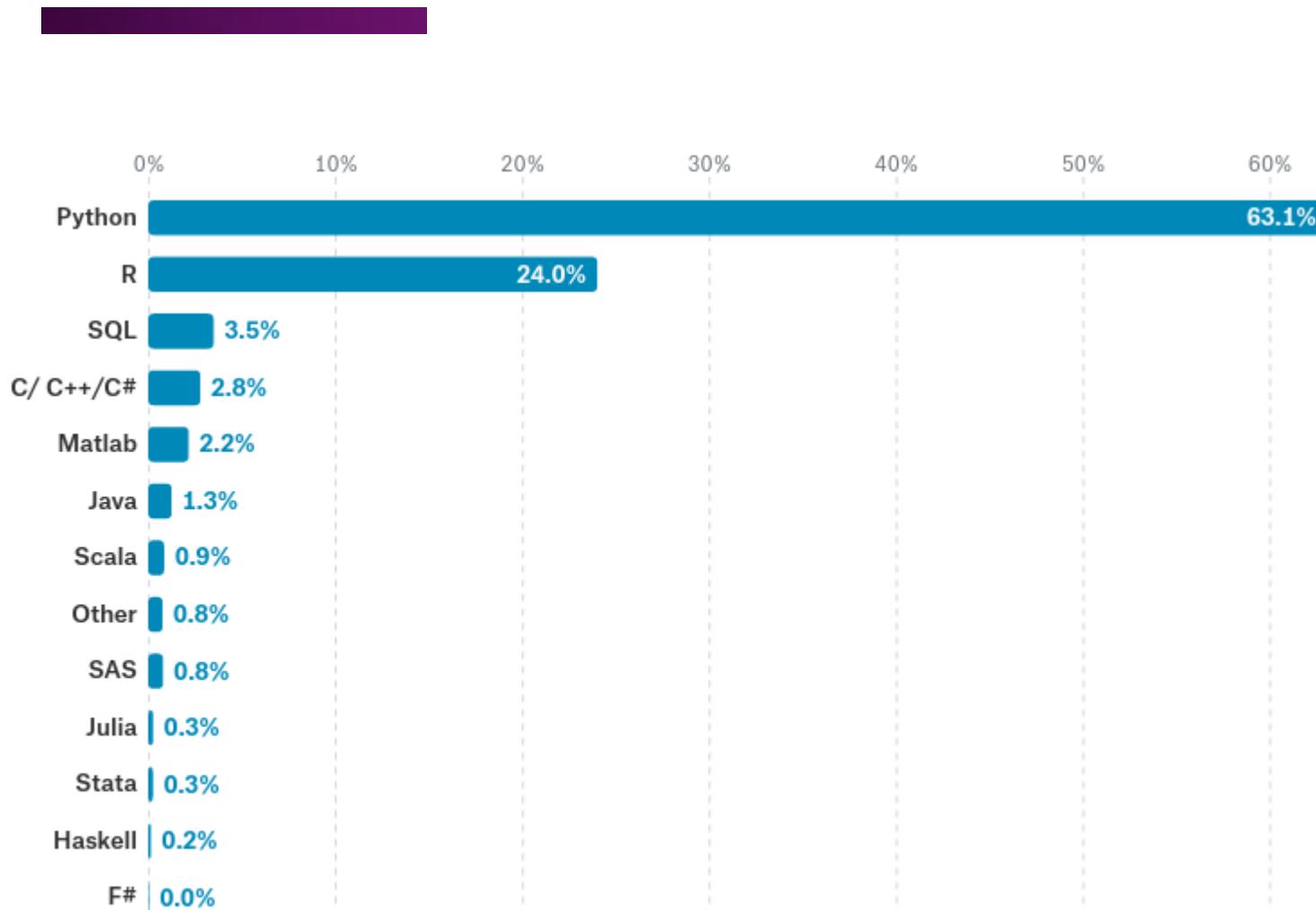


Lenguajes de programación

Lenguaje de programación es una estructura que, con una base sintáctica y semántica, indica distintas instrucciones a un programa de Computador.



Lenguajes de programación



Un dato muy significativo es la respuesta a la pregunta: *¿Qué lenguaje de programación recomendaría a los nuevos científicos de datos aprender primero?* Más del 63% de los encuestados respondió Python.

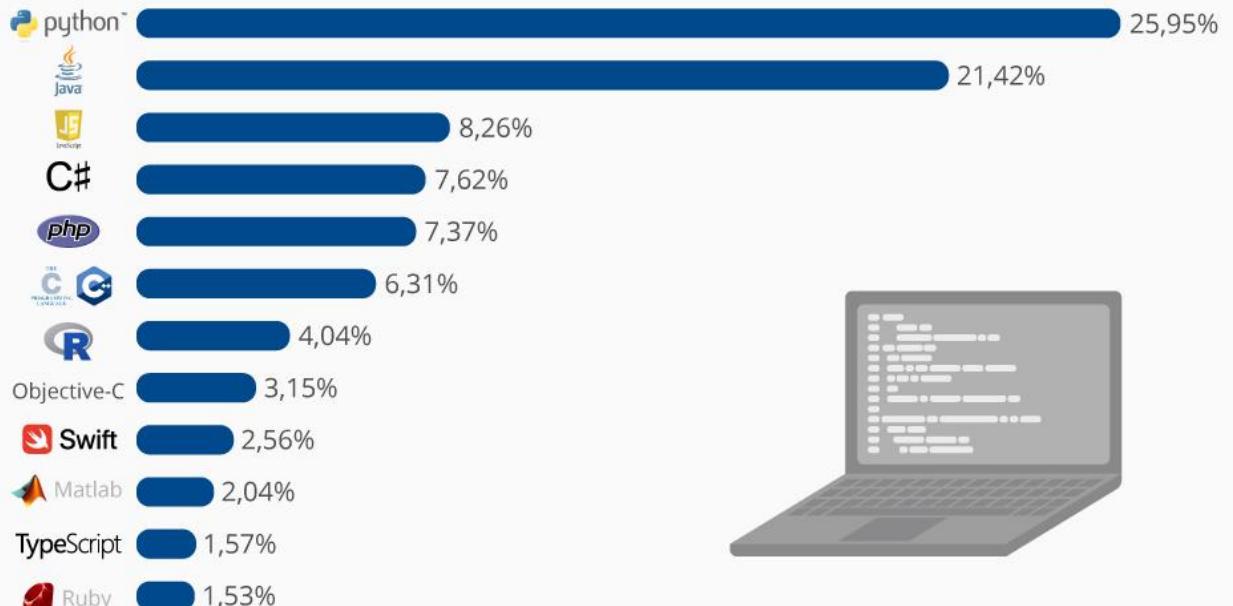
Lenguajes de programación

Hay que diferenciar entre:

- **Lenguaje de programación**
 - Java
 - C#
 - Python
- **Lenguaje informático.**
 - HTML (un lenguaje de marcas).

Los lenguajes de programación más usados

Porcentaje de uso de los lenguajes de programación más populares del mundo*



* Datos procedentes del Índice PYPL. Este se basa en las búsquedas en Google de tutoriales de lenguajes de programación.

Datos de enero de 2019

Fuente: PYPL



Lenguajes de programación

NIVEL DE ABSTRACCIÓN

lenguaje de máquina

Cadenas binarias que pueden ser legibles de manera directa por la computadora
101011101001

“El lenguaje es fuente
de malos entendidos”

El principito.



lenguaje de bajo nivel

El lenguaje de programación que se acerca al funcionamiento de un ordenador
Ensamblador

lenguaje de medio nivel

Se encuentran entre los lenguajes de alto nivel y los lenguajes de bajo nivel.

C

lenguaje de alto nivel

Formado por elementos del lenguaje humano

C++

Fortran.

Java.

PHP.

Python.

Lenguajes de programación

11001010 00010111 11110101 00101011
00010111 11110101 00101011 00101011
11001010 00010111 11110101 00101011
00010111 11110101 00101011 00101011
11001010 11110101 00101011 00101011
11001010 11001010 11110101 00101011
11001010 11110101 00101011 00101011
11001010 00010111 11110101 00101011
00010111 11110101 00101011 00101011
11001010 11110101 00101011 00101011

LENGUAJE MÁQUINA

“El lenguaje es fuente
de malos entendidos”

El principito.



Lenguajes de programación

Ejemplo para la arquitectura x86 [\[editar\]](#)

El siguiente es un ejemplo del programa clásico *Hola mundo* escrito para la arquitectura de procesador x86 (bajo el sistema operativo DOS).

```
; -----
; Programa que imprime un string en la pantalla
;
.model small ; modelo de memoria

.stack ; segmento del stack

.data ; segmento de datos
Cadena1 DB 'Hola Mundo.$' ; string a imprimir (finalizado en $)

.code ; segmento del código

;
; Inicio del programa
;

programa:
;
; inicia el segmento de datos
;
MOV AX, @data ; carga en AX La dirección del segmento de datos
MOV DS, AX ; mueve La dirección al registro de segmento por medio de AX

;
; Imprime un string en pantalla
;
MOV DX, offset Cadena1 ; mueve a DX la dirección del string a imprimir
MOV AH, 9 ; AH = código para indicar al MS DOS que imprima en la pantalla, el string en DS:DX
INT 21h ; Llamada al MS DOS para ejecutar la función (en este caso especificada en AH)

;
; Finaliza el programa
;
INT 20h ; Llamada al MS DOS para finalizar el programa

end programa
```

LENGUAJE ENSAMBLADOR

“El lenguaje es fuente de malos entendidos”

El principito,



Lenguajes de programación

ejemplo C: Hola Mundo !

```
#include <stdio.h>

int main()
{
    printf("Hola Mundo! \n");
    return 0;
}
```

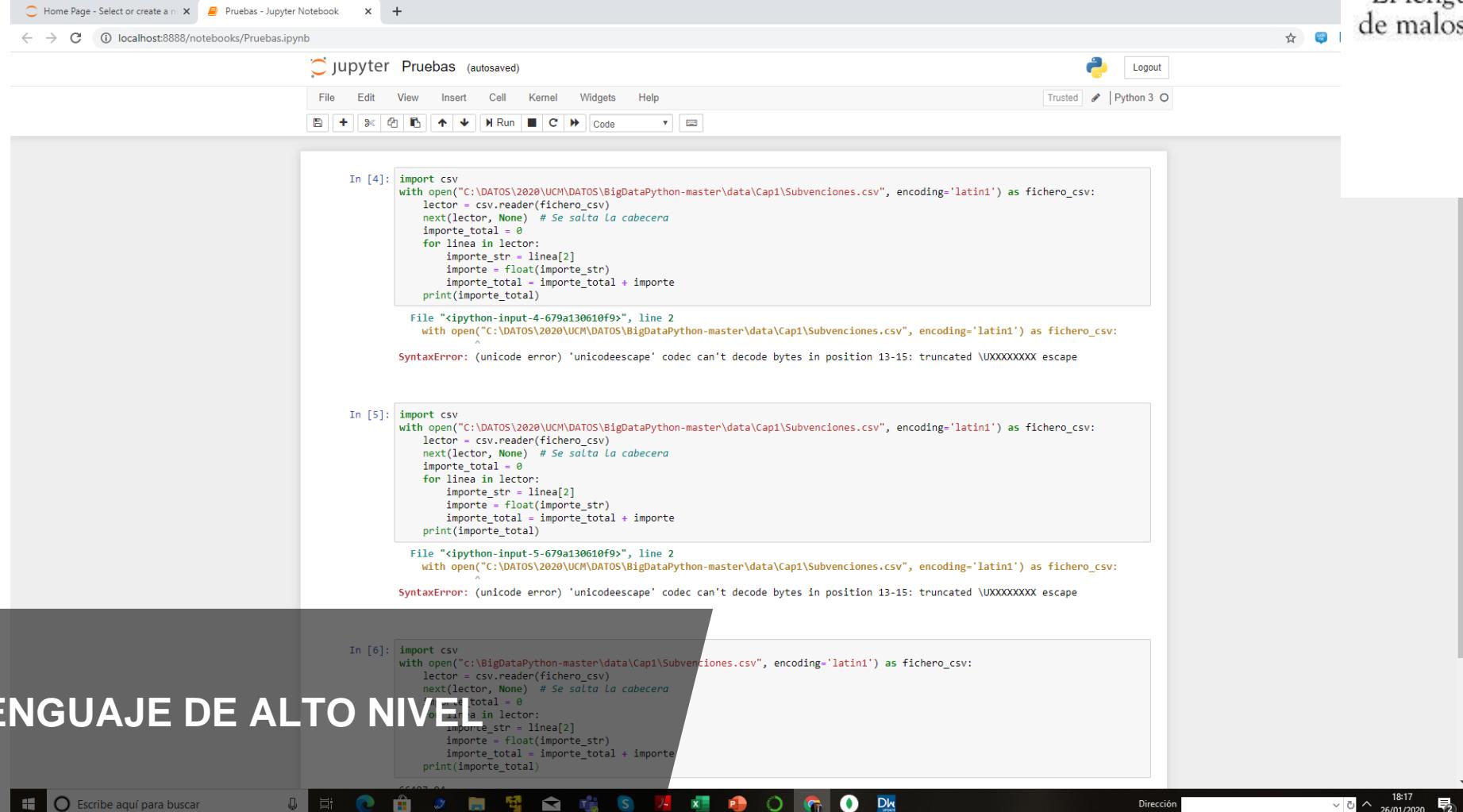
"El lenguaje es fuente
de malos entendidos"

El principito.



LENGUAJE DE MEDIO NIVEL

Lenguajes de programación



In [4]:

```
import csv
with open("C:\DATOS\2020\UCM\DATOS\BigDataPython-master\data\Cap1\Subvenciones.csv", encoding='latin1') as fichero_csv:
    lector = csv.reader(fichero_csv)
    next(lector, None) # Se salta la cabecera
    importe_total = 0
    for linea in lector:
        importe_str = linea[2]
        importe = float(importe_str)
        importe_total = importe_total + importe
    print(importe_total)

File "<ipython-input-4-679a130610f9>", line 2
  with open("C:\DATOS\2020\UCM\DATOS\BigDataPython-master\data\Cap1\Subvenciones.csv", encoding='latin1') as fichero_csv:
                                         ^
SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in position 13-15: truncated \UXXXXXXXXX escape
```

In [5]:

```
import csv
with open("C:\DATOS\2020\UCM\DATOS\BigDataPython-master\data\Cap1\Subvenciones.csv", encoding='latin1') as fichero_csv:
    lector = csv.reader(fichero_csv)
    next(lector, None) # Se salta la cabecera
    importe_total = 0
    for linea in lector:
        importe_str = linea[2]
        importe = float(importe_str)
        importe_total = importe_total + importe
    print(importe_total)

File "<ipython-input-5-679a130610f9>", line 2
  with open("C:\DATOS\2020\UCM\DATOS\BigDataPython-master\data\Cap1\Subvenciones.csv", encoding='latin1') as fichero_csv:
                                         ^
SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in position 13-15: truncated \UXXXXXXXXX escape
```

In [6]:

```
import csv
with open("c:\BigDataPython-master\data\Cap1\Subvenciones.csv", encoding='latin1') as fichero_csv:
    lector = csv.reader(fichero_csv)
    next(lector, None) # Se salta la cabecera
    importe_total = 0
    for linea in lector:
        importe_str = linea[2]
        importe = float(importe_str)
        importe_total = importe_total + importe
    print(importe_total)
```

LENGUAJE DE ALTO NIVEL

"El lenguaje es fuente
de malos entendidos"

El principito,



Lenguajes de programación

LENGUAJES COMPILADOS E INTERPRETADOS

Lenguaje Compilado

Tras escribir el programa debe ser pasado por un “programa especial” (compilador) que lo lee y crea a partir de él una versión en código máquina entendible por el procesador.

El código escrito es el código fuente y la versión compilada el binario.

Si modificamos el código fuente es necesario volver a compilar.

Es mas rápido que el lenguaje interpretado.

Funcionan sin la necesidad de un programa que los interprete.

Cada versión que se compila depende de la plataforma en que lo queramos usar.

Ejemplos: C , C++, C# ...

“El lenguaje es fuente
de malos entendidos”

El principito,



Lenguajes de programación

LENGUAJES COMPILADOS E INTERPRETADOS

Lenguaje Interpretado

Se ejecuta sin compilar.

En lugar de compilador se usa un intérprete.

No existe binario.

El programa escrito se llama script.

No dependen de la plataforma si se dispone del intérprete.

Suelen ser mas lentos que los compilados.

Perl, Ruby, Python...

“El lenguaje es fuente
de malos entendidos”

El principito,



Python

“Es un lenguaje **interpretado**, de **alto** nivel y enfocado principalmente a la legibilidad y facilidad de uso.”



“**Python** es un lenguaje de **programación** multiparadigma. Esto significa que más que forzar a los programadores a adoptar un estilo particular de **programación**, permite varios estilos: **programación** orientada a objetos, **programación** imperativa y **programación** funcional.”

Python

Características

Tipado Dinámico

```
In [1]: def saludo(nombre):
           return 'Hola {}'.format(nombre)
```

```
In [2]: print (saludo('Raul'))
           print (saludo(1))
```

```
Hola Raul
Hola 1
```

```
In [3]: def saludo(nombre):
           if type(nombre) != str:
               return "Error: el argumento debe ser del tipo String(str)"
           return 'Hola {}'.format(nombre)
```

Se dice de un lenguaje de programación que usa un **tipado dinámico** cuando la comprobación de tipificación se realiza durante su ejecución en vez de durante la compilación.

```
print(saludo('Raul'))
print(saludo(1))
Hola Raul
Error: el argumento debe ser del tipo String(str)
```

Python

Características

Uso de bibliotecas

Software libre. Licencia “Python Software Foundation License”, se distribuye gratuitamente, no se pagan licencias.

Creador: Guido van Rossum.

Filosofía: Código limpio y legible. Simple sin ser limitado.

Gran cantidad de paquetes (pilas incluidas)

Case sensitive

The following code is an example of case insensitive string comparison in Python.

Example

```
string1 = 'Star Wars'
string2 = 'star wars'
if string1.lower() == string2.lower():
    print "The strings are case insensitive"
else:
    print "The strings are not case insensitive"
```

Output

This code gives the following output

```
The strings are case insensitive
```



Now, it's my belief that Python is a lot easier than to teach to students programming and teach them C or C++ or Java at the same time because all the details of the languages are so much harder. Other scripting languages really don't work very well there either.

- Guido van Rossum -

quoteparrot.com

Python

The Zen of Python

```
Beautiful is better than ugly.  
Explicit is better than implicit.  
Simple is better than complex.  
Complex is better than complicated.  
Flat is better than nested.  
Sparse is better than dense.  
Readability counts.  
Special cases aren't special enough to break the rules.  
Although practicality beats purity.  
Errors should never pass silently.  
Unless explicitly silenced.  
In the face of ambiguity, refuse the temptation to guess.  
There should be one-- and preferably only one --obvious way to do it.  
Although that way may not be obvious at first unless you're Dutch.  
Now is better than never.  
Although never is often better than *right* now.  
If the implementation is hard to explain, it's a bad idea.  
If the implementation is easy to explain, it may be a good idea.  
Namespaces are one honking great idea -- let's do more of those!
```



Python

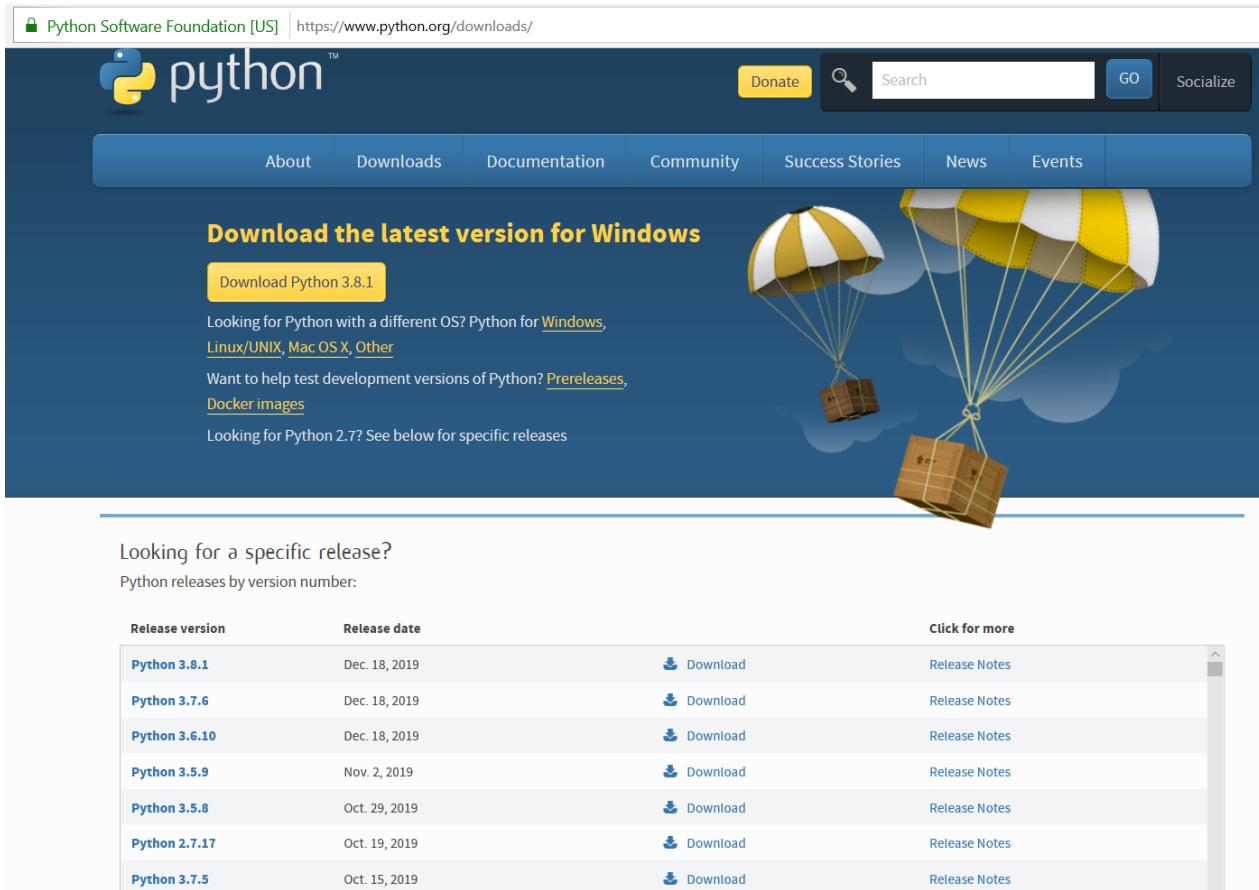
Hermoso es mejor que feo.
Explícito es mejor que implícito.
Simple es mejor que complejo.
Complejo es mejor que complicado.
Plano es mejor que anidado.
Escaso es mejor que denso.
La legibilidad cuenta.
Los casos especiales no son lo suficientemente especiales como para romper las reglas.
Aunque la practicidad supera la pureza.
Los errores nunca deben pasar en silencio.
A menos que sea silenciado explícitamente.
Ante la ambigüedad, rechaza la tentación de adivinar.
Debe haber una, y preferiblemente solo una, forma obvia de hacerlo.
Aunque esa manera puede no ser obvia al principio a menos que seas holandés.
Ahora es mejor que nunca.
Aunque nunca es mejor que el ahora correcto.
Si la implementación es difícil de explicar, es una mala idea.
Si la implementación es fácil de explicar, puede ser una buena idea.
Los espacios de nombres son una gran idea, ¡hagamos más de eso!



Python

Versiones

Enero 2020: Python 2.7.17 y Python 3.8.1



The screenshot shows the Python Software Foundation's Downloads page. At the top, there is a navigation bar with links for About, Downloads, Documentation, Community, Success Stories, News, and Events. A prominent yellow button invites users to "Download Python 3.8.1". Below this, there are links for other operating systems and development tools. A large graphic of two packages descending from the sky on parachutes is centered on the page. At the bottom, a table lists previous Python releases:

Release version	Release date	Click for more
Python 3.8.1	Dec. 18, 2019	Download Release Notes
Python 3.7.6	Dec. 18, 2019	Download Release Notes
Python 3.6.10	Dec. 18, 2019	Download Release Notes
Python 3.5.9	Nov. 2, 2019	Download Release Notes
Python 3.5.8	Oct. 29, 2019	Download Release Notes
Python 2.7.17	Oct. 19, 2019	Download Release Notes
Python 3.7.5	Oct. 15, 2019	Download Release Notes

Python

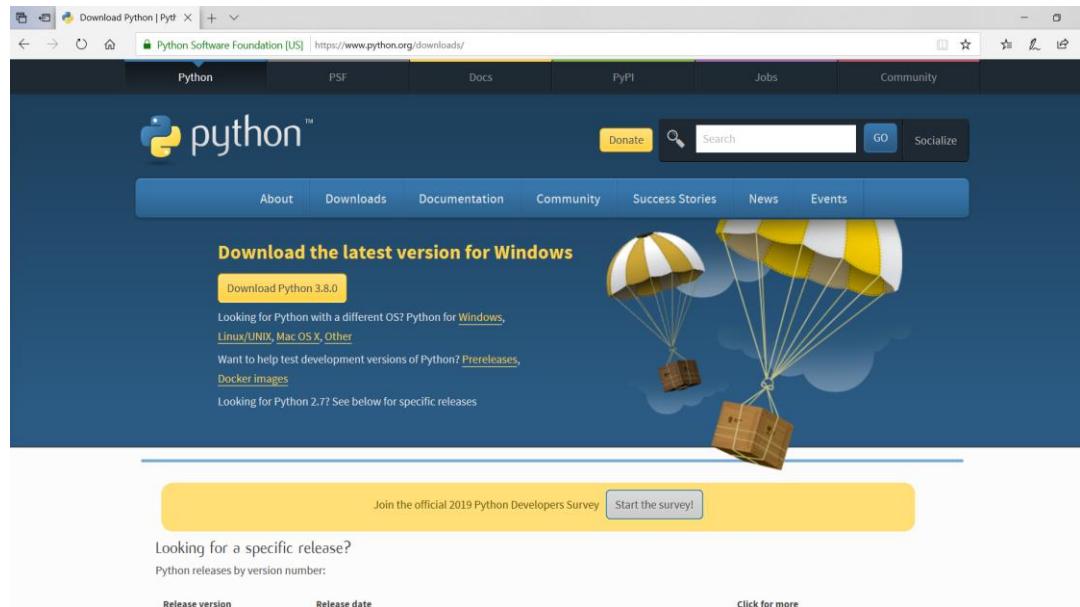
Principales diferencias entre versiones

- Python 3 resuelve muchos problemas de codificación de texto, y ahora todas las cadenas son Unicode por defecto.
- Todas las clases de Python 3 son New Style. No existen las clases Old Style.
- En Python 3, print es una función y no una sentencia del lenguaje, por lo que debe usarse con paréntesis. Ejemplo: print("texto") en lugar de print "texto".
- En Python 3 las funciones filter(), map() o zip() devuelven un iterable, antes devolvían listas.
- Python 2, división de enteros devuelve entero. En Python 3 devuelve en coma flotante.
- Xrange() de Python 2 se convierte en range()
- Raw_input() se convierte en input(). Devuelve objeto string.
- Next() desaparece.
- Python 3 devuelve Type Error al intentar comparar tipos no ordenables.

Python

Instalación Python 3.8.0 (Noviembre 2019)

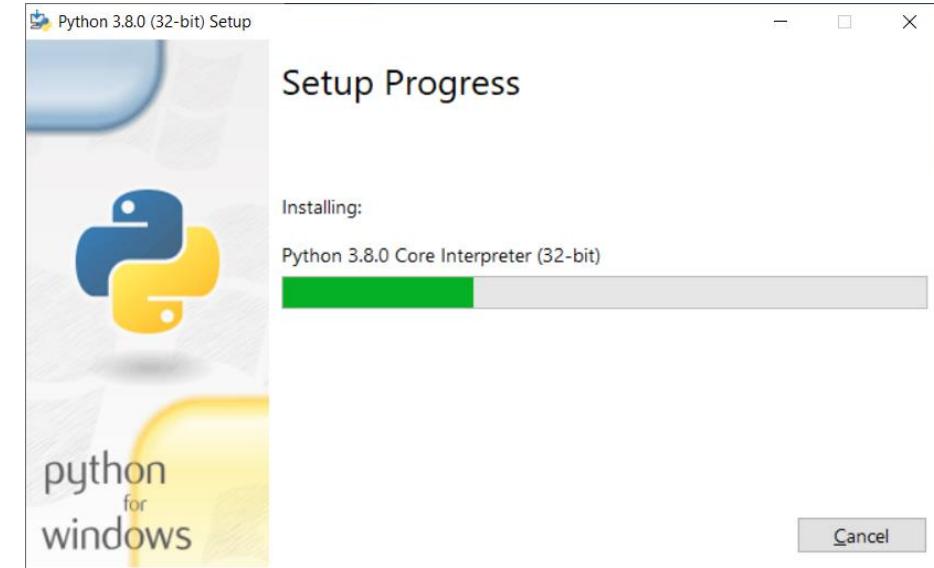
<https://www.python.org/downloads/>



Python

Instalación Python 3.8.0 (Noviembre 2019)

Ojo, marcar “Add Python 3.8 to PATH”

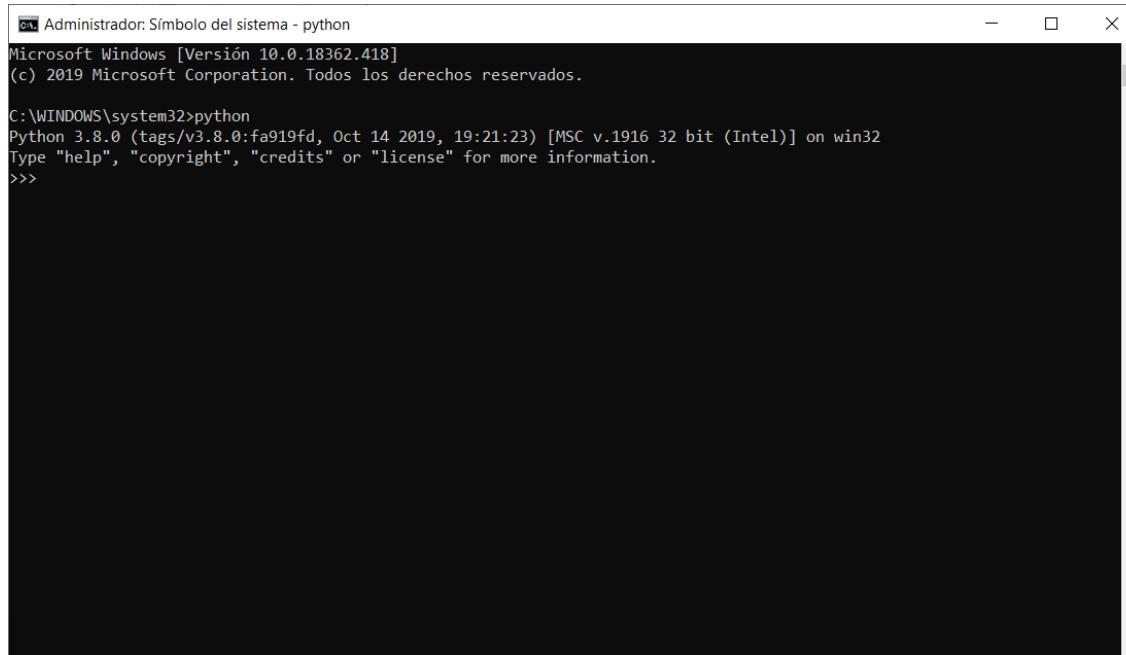


Python

Instalación Python 3.8.0 (Noviembre 2019)

Probar que funciona

Desde ms-dos (preferible ejecutar como administrador), escribir “python” y pulsar intro.



```
Administrator: Símbolo del sistema - python
Microsoft Windows [Versión 10.0.18362.418]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\WINDOWS\system32>python
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:21:23) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

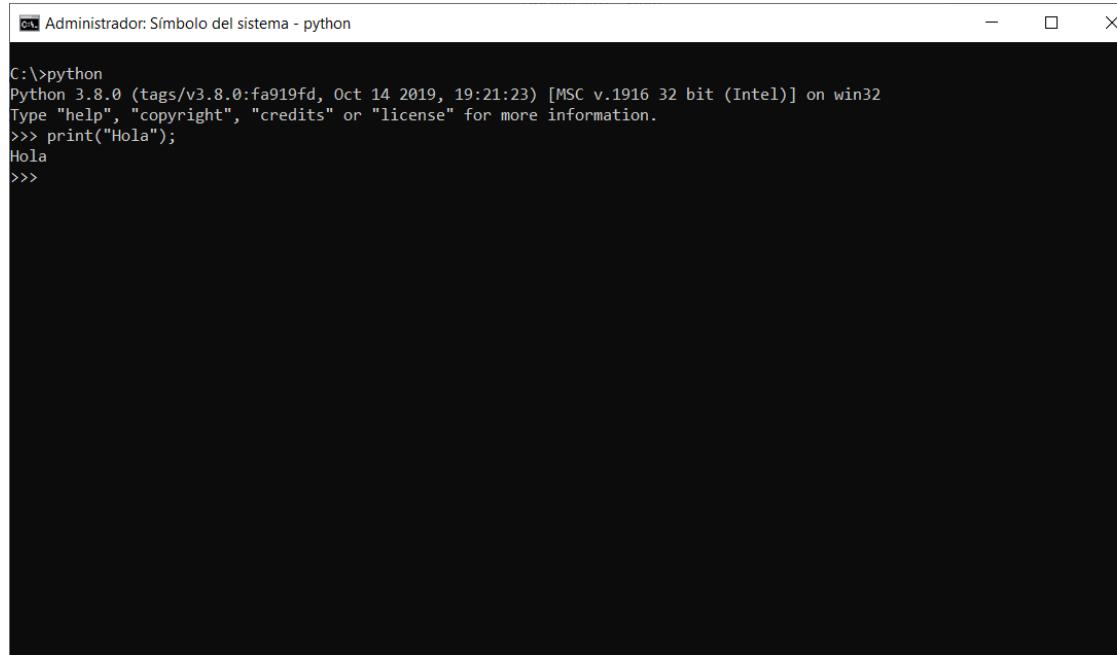
Python

Instalación Python 3.8.0 (Noviembre 2019)

Probar que funciona

Escribir Python y pulsar intro.

Escribir print("Hola"); y pulsar intro.



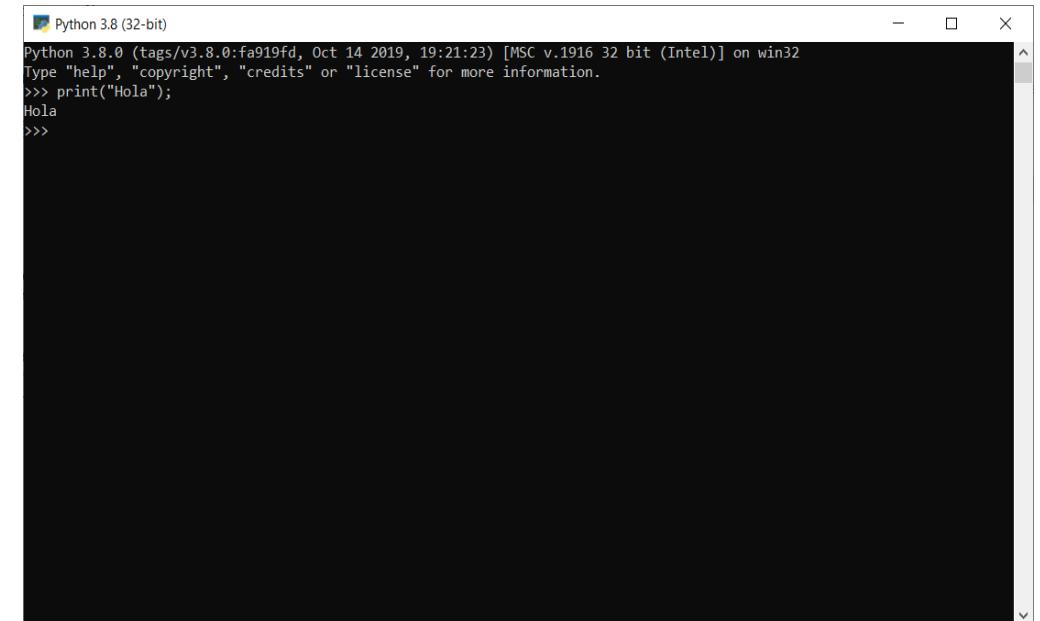
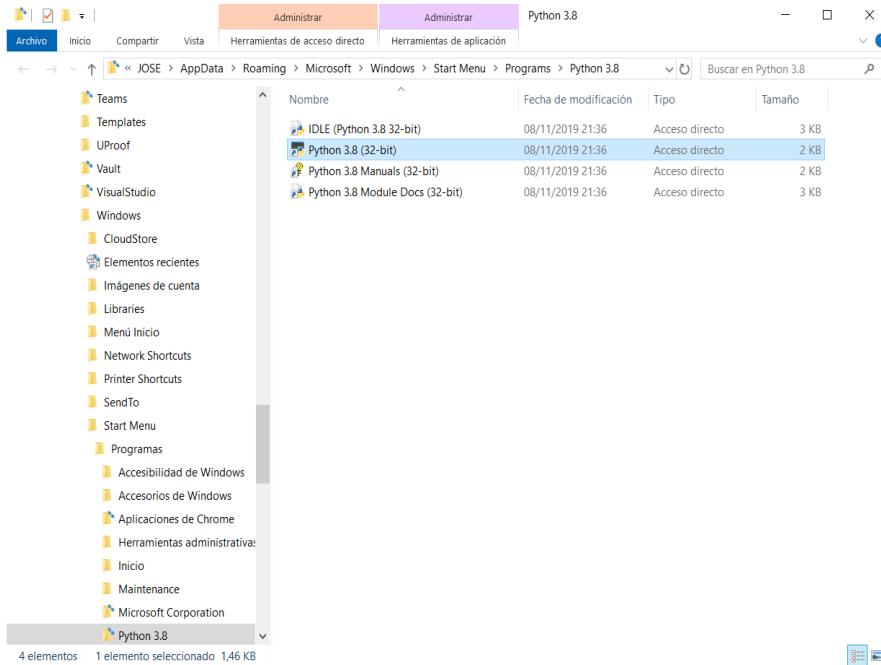
```
C:\>python
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:21:23) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hola")
Hola
>>>
```

Python

Instalación Python 3.8.0 (Noviembre 2019)

Probar que funciona

También se puede probar usando el ejecutable instalado:



A screenshot of a Python 3.8 terminal window titled "Python 3.8 (32-bit)". The window shows the Python interpreter prompt ">>>". The user typed "print("Hola")" and the terminal responded with "Hola". The window also displays the Python version information at the top: "Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:21:23) [MSC v.1916 32 bit (Intel)] on win32".

Python

PIP

Sistema gestor de paquetes para instalar y administrar programas hechos en Python

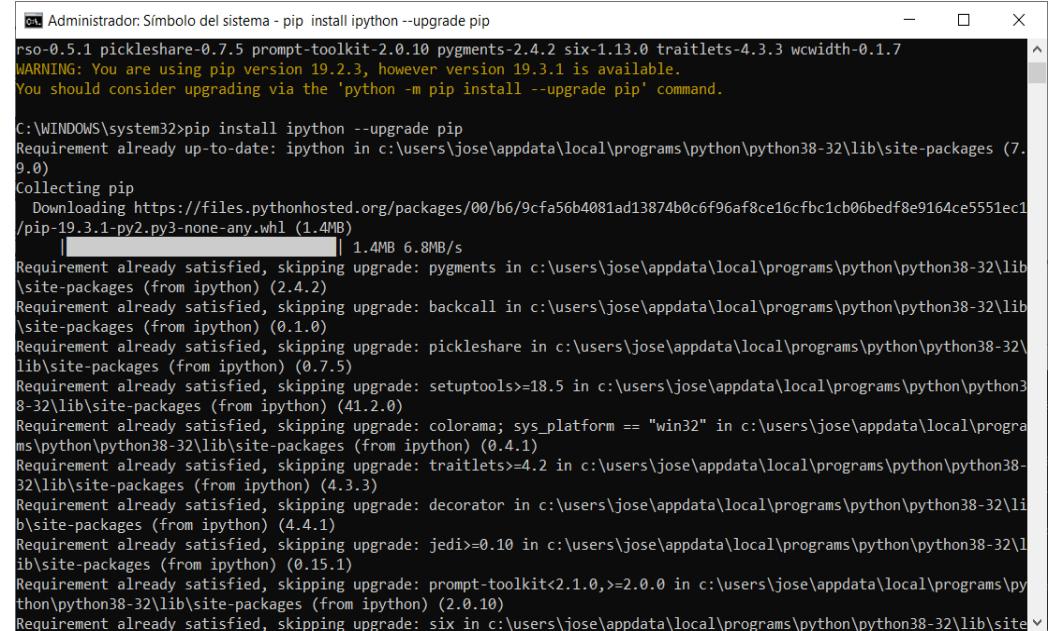
Ejecutar ms-dos como administrador.

Instalar PIP*

python setup.py install

Actualizar PIP

Pip install ipython –upgrade pip



```
Administrator: Símbolo del sistema - pip install ipython --upgrade pip
rso-0.5.1 pickleshare-0.7.5 prompt-toolkit-2.0.10 pygments-2.4.2 six-1.13.0 traitlets-4.3.3 wcwidth-0.1.7
WARNING: You are using pip version 19.2.3, however version 19.3.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\WINDOWS\system32>pip install ipython --upgrade pip
Requirement already up-to-date: ipython in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (7.9.0)
Collecting pip
  Downloading https://files.pythonhosted.org/packages/00/b6/9cfaf56b4081ad13874b0c6f96af8ce16cfbc1cb06bedf8e9164ce5551ec1/pip-19.3.1-py2.py3-none-any.whl (1.4MB)
    |████████| 1.4MB 6.8MB/s
Requirement already satisfied, skipping upgrade: pygments in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (2.4.2)
Requirement already satisfied, skipping upgrade: backcall in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (0.1.0)
Requirement already satisfied, skipping upgrade: pickleshare in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (0.7.5)
Requirement already satisfied, skipping upgrade: setuptools>=18.5 in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (41.2.0)
Requirement already satisfied, skipping upgrade: colorama; sys_platform == "win32" in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (0.4.1)
Requirement already satisfied, skipping upgrade: traitlets>=4.2 in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (4.3.3)
Requirement already satisfied, skipping upgrade: decorator in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (4.4.1)
Requirement already satisfied, skipping upgrade: jedi>=0.10 in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (0.15.1)
Requirement already satisfied, skipping upgrade: prompt-toolkit<2.1.0,>=2.0.0 in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (2.0.10)
Requirement already satisfied, skipping upgrade: six in c:\users\jose\appdata\local\programs\python\python38-32\lib\site-packages (from ipython) (1.14.0)
```

* Sistema de gestión de paquetes utilizado para instalar y administrar paquetes de software escritos en Python 2.7.9 y posteriores

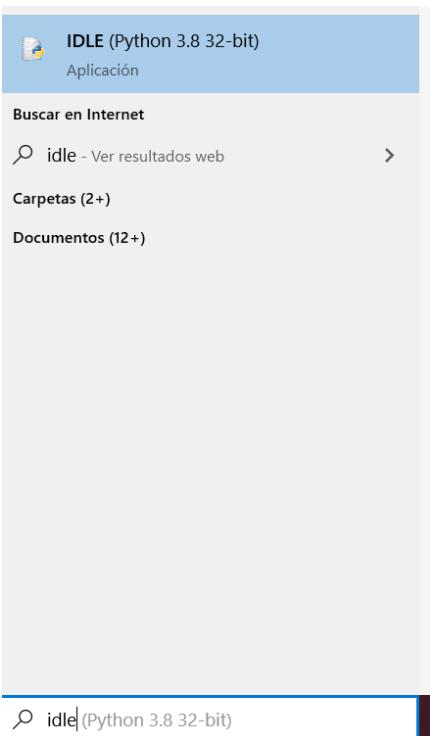
Python

Editor de texto

IDLE

Instalado por defecto con Python en Windows.

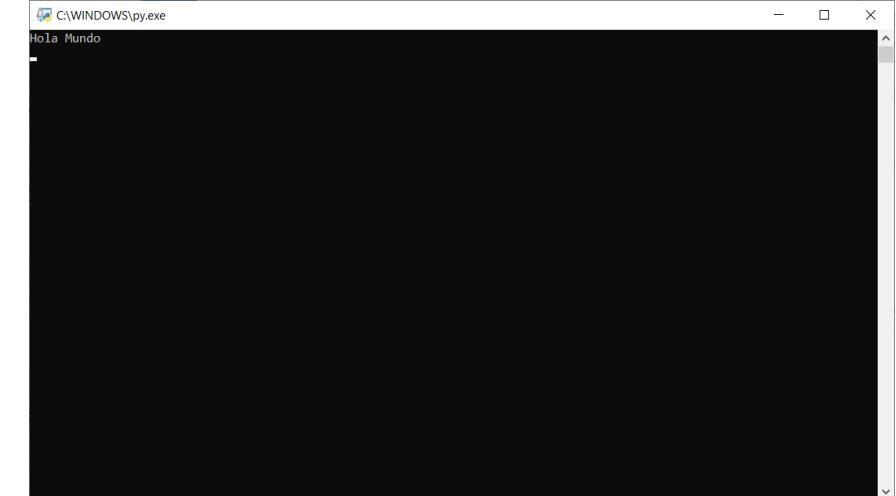
Ejemplo: Abrimos editor escribimos un programa y pulsamos doble clic sobre el archivo guardado.



The screenshot shows the Python IDLE editor window. The title bar reads 'Hola.py - C:\EjerciciosPython\Hola.py (3.8.0)'. The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code area contains the following Python code:

```
print ("Hola Mundo")
input()
```

The status bar at the bottom right shows 'Ln: 1 Col: 0'.



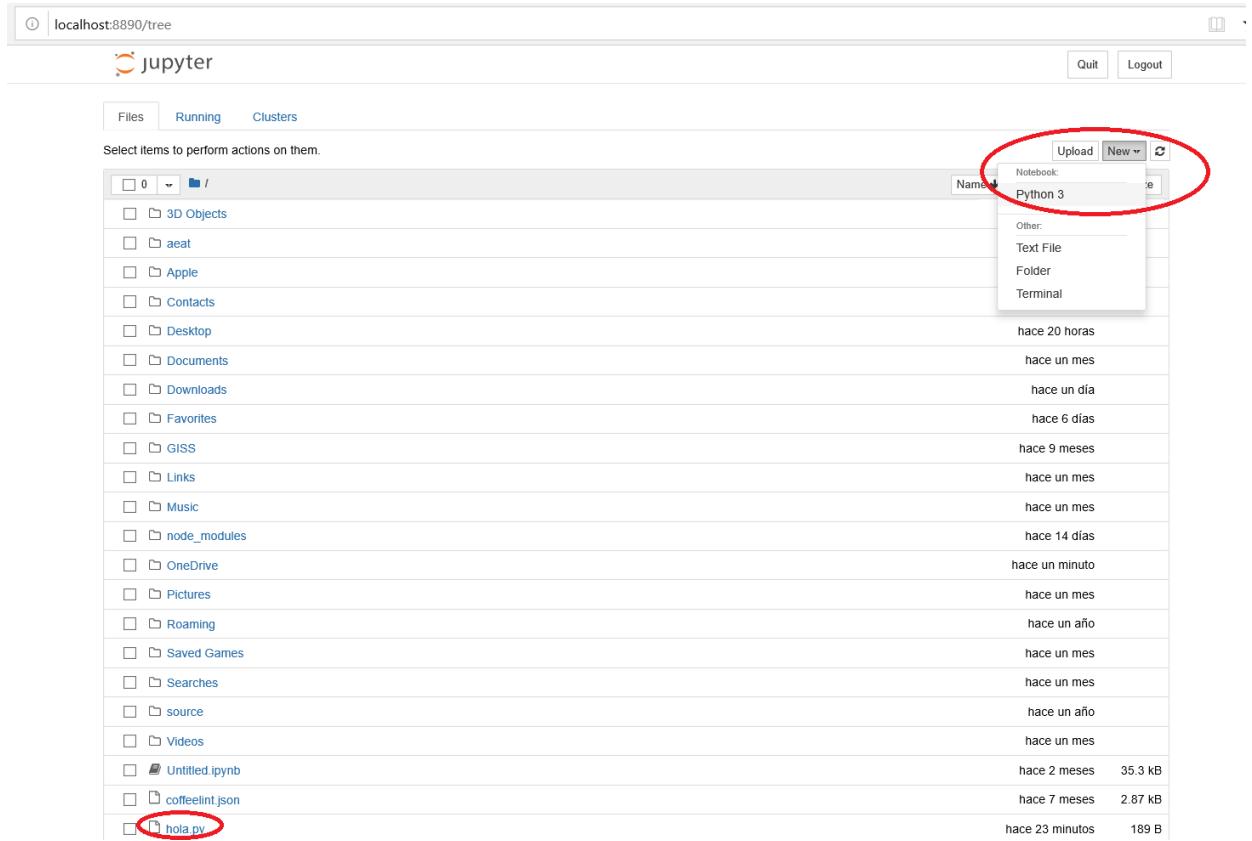
Python

Editor de texto

JUPYTER (También se puede usar a través de Anaconda)

Instalación: Python –m pip install jupyter

```
C:\WINDOWS\system32>python -m pip install jupyter
Collecting jupyter
  Downloading https://files.pythonhosted.org/packages/83/df/0f5dd132200728a86190397e1ea87cd76244e42d39ec5e88efd25b2abd7e
/jupyter-1.0.0-py2.py3-none-any.whl
Collecting notebook
  Downloading https://files.pythonhosted.org/packages/f5/69/d2ffaf7efc20ce47469187e3a41e6e03e17b45de5a6559f4e7ab3eace5e1
/notebook-6.0.2-py3-none-any.whl (9.7MB)
|██████████| 9.7MB 6.4MB/s
```

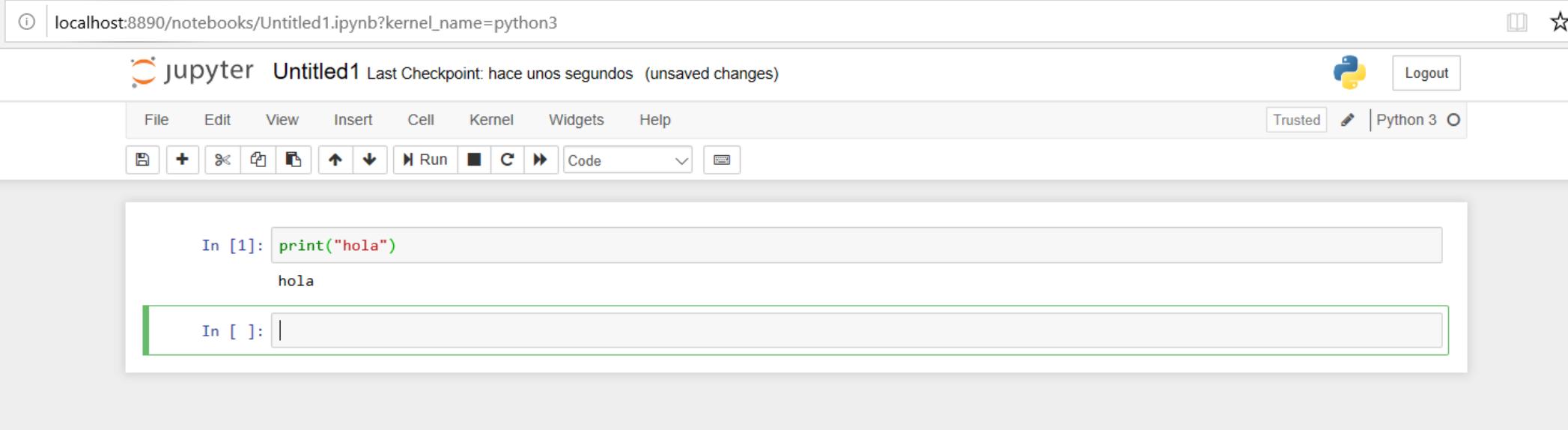


Python



Editor de texto

JUPYTER



The screenshot shows a Jupyter Notebook interface. The title bar indicates the URL is `localhost:8890/notebooks/Untitled1.ipynb?kernel_name=python3`. The main window displays a single code cell with the following content:

```
In [1]: print("hola")
```

The output of the cell is:

```
hola
```

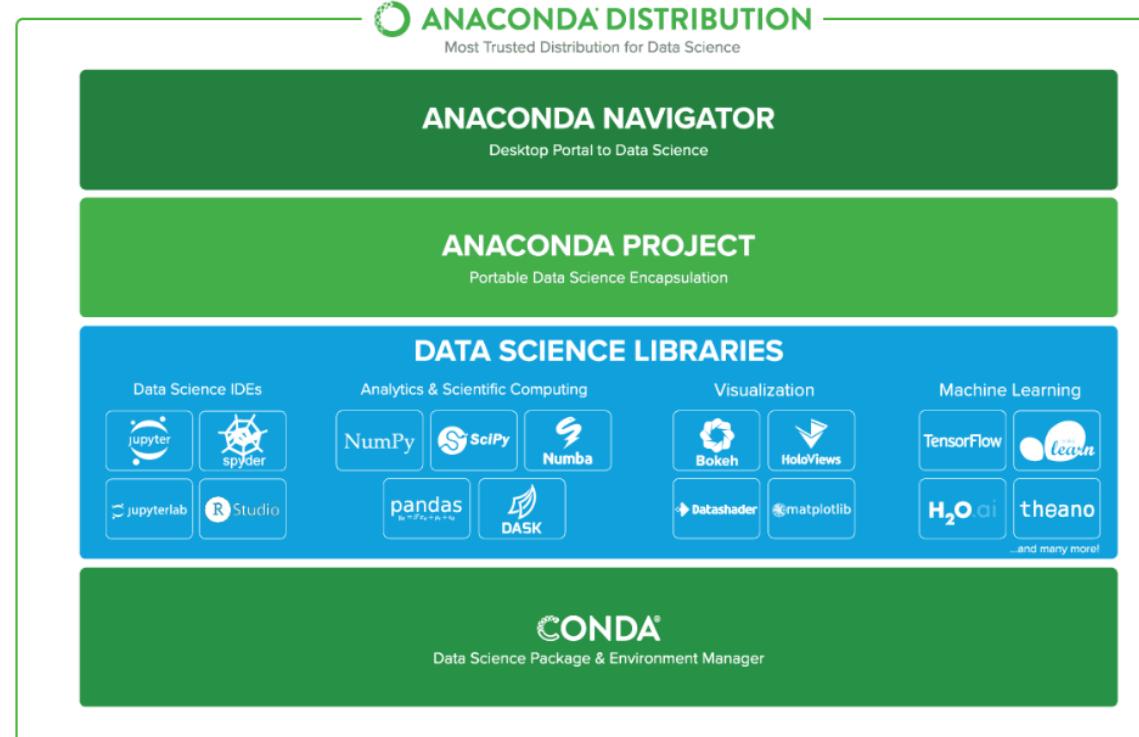
A new code cell is currently being edited, indicated by the cursor in the input field:

```
In [ ]: |
```

Python

ANACONDA

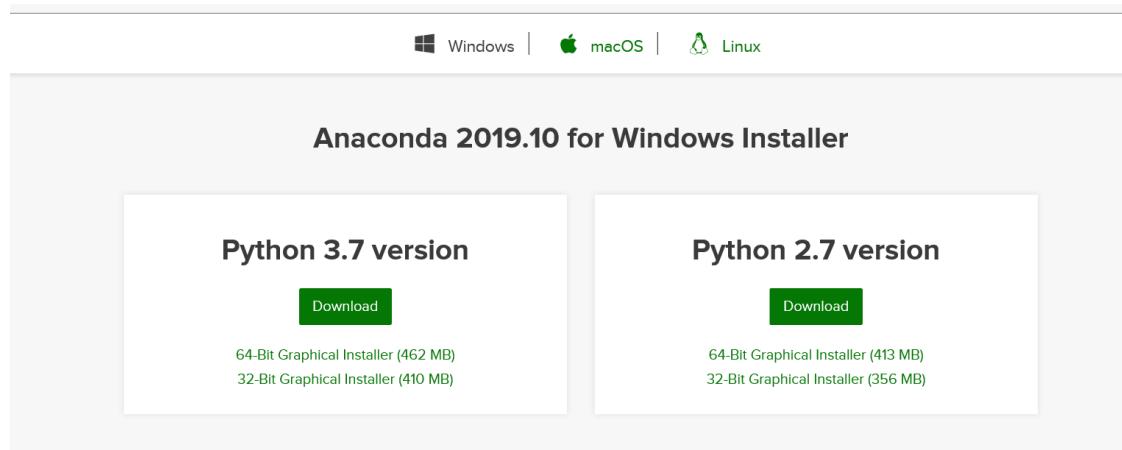
Anaconda es una Suite de código abierto que abarca una serie de aplicaciones, librerías y conceptos diseñados para el desarrollo de la Ciencia de datos con Python. En líneas generales Anaconda Distribution es una distribución de Python que funciona como un gestor de entorno, un gestor de paquetes y que posee una colección de más de 720 paquetes de código abierto.



Python

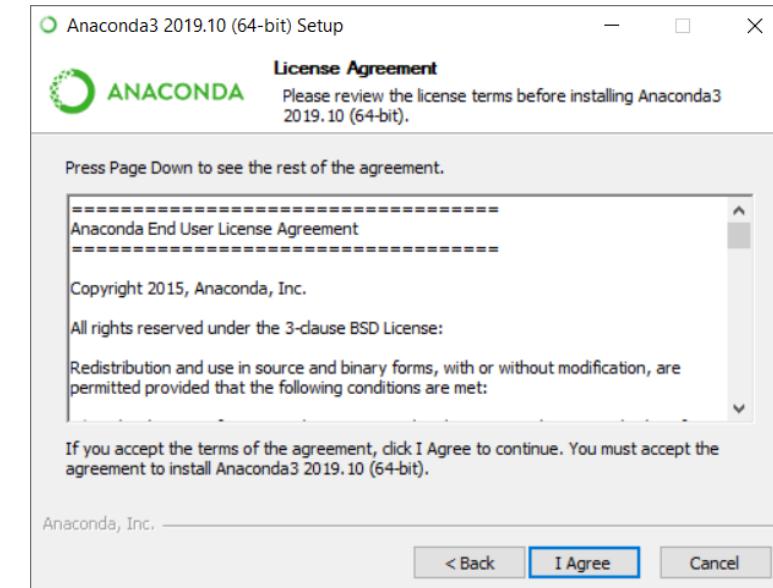
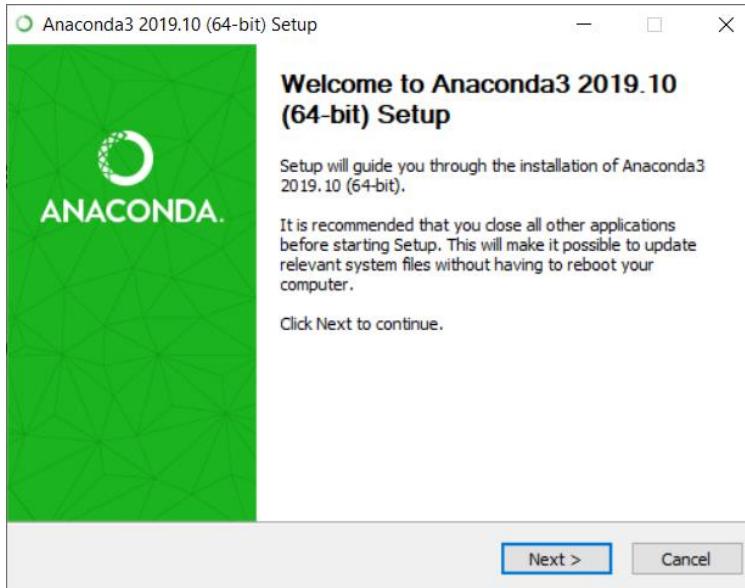
ANACONDA

<https://www.anaconda.com/distribution/#download-section>



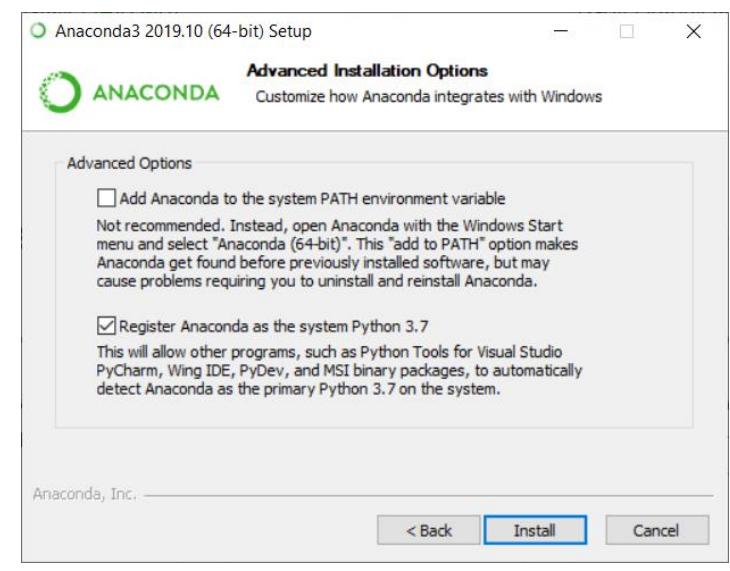
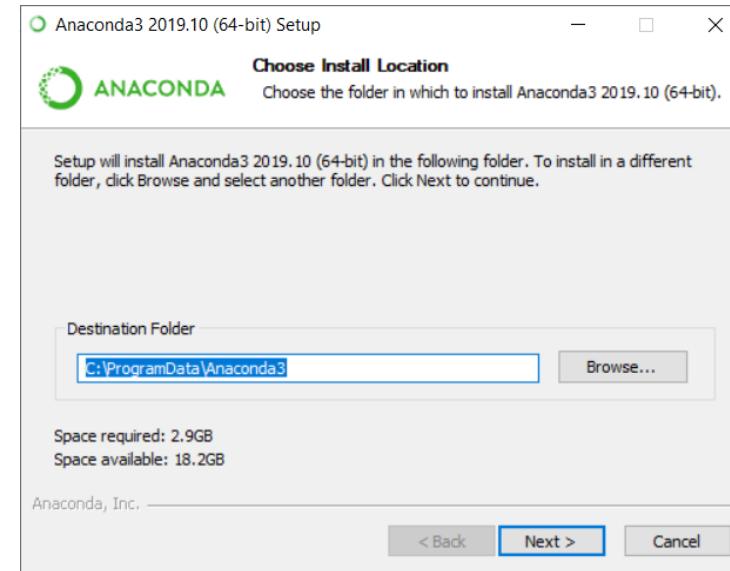
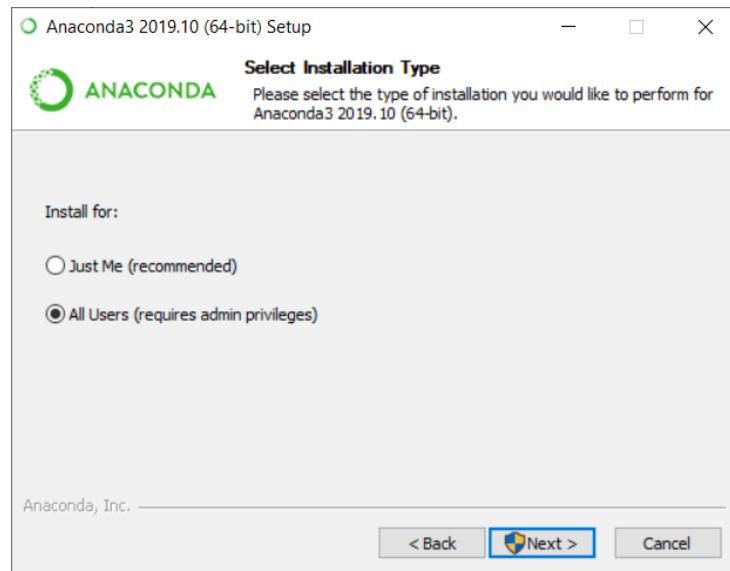
Python

ANACONDA

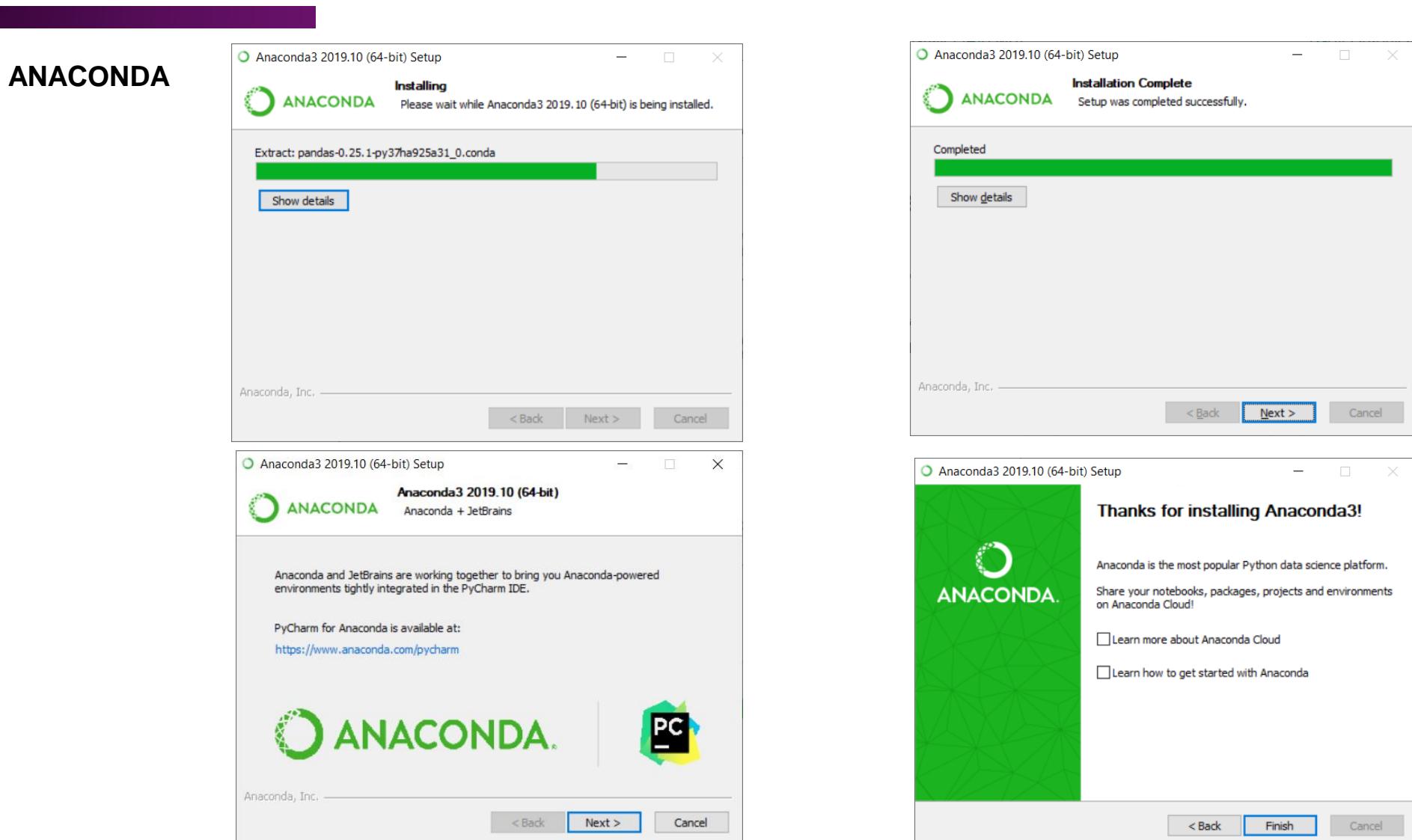


Python

ANACONDA

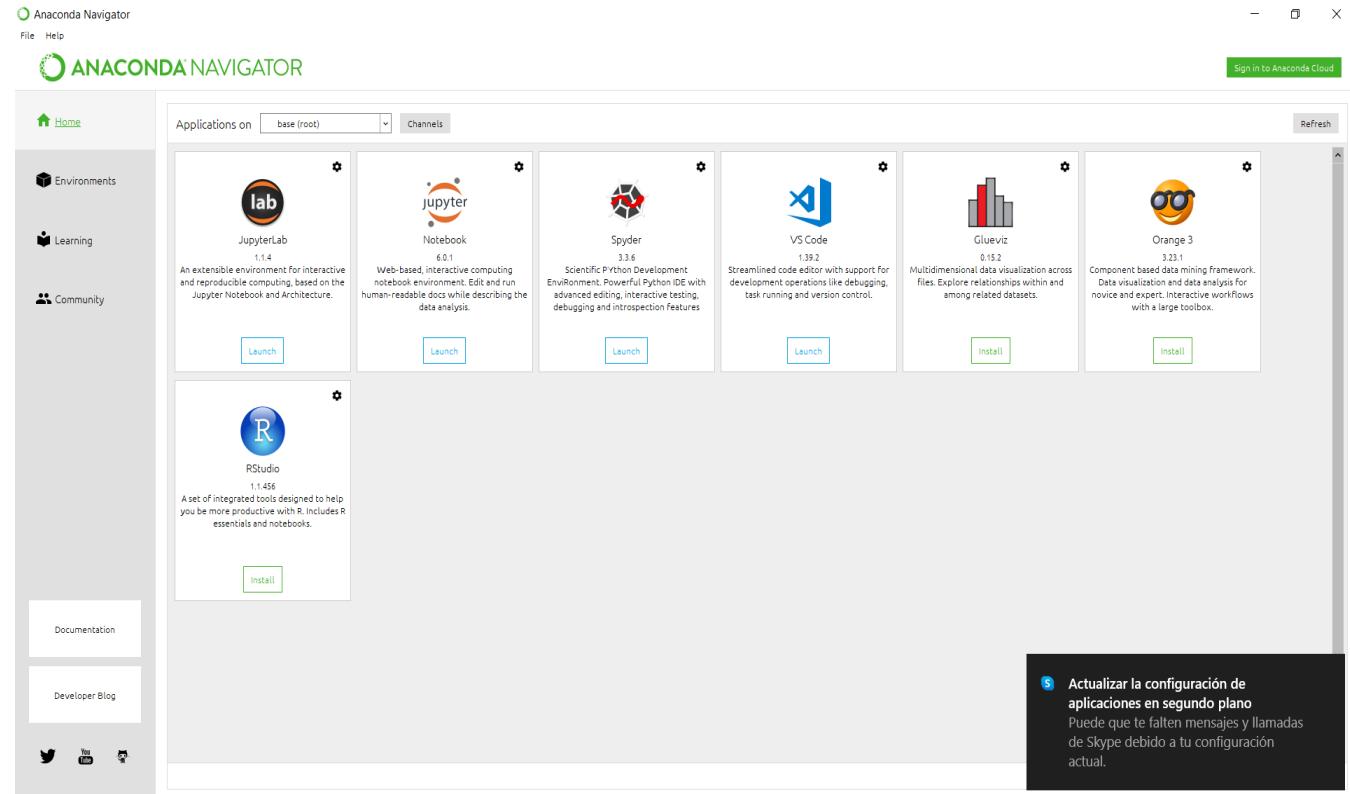


Python



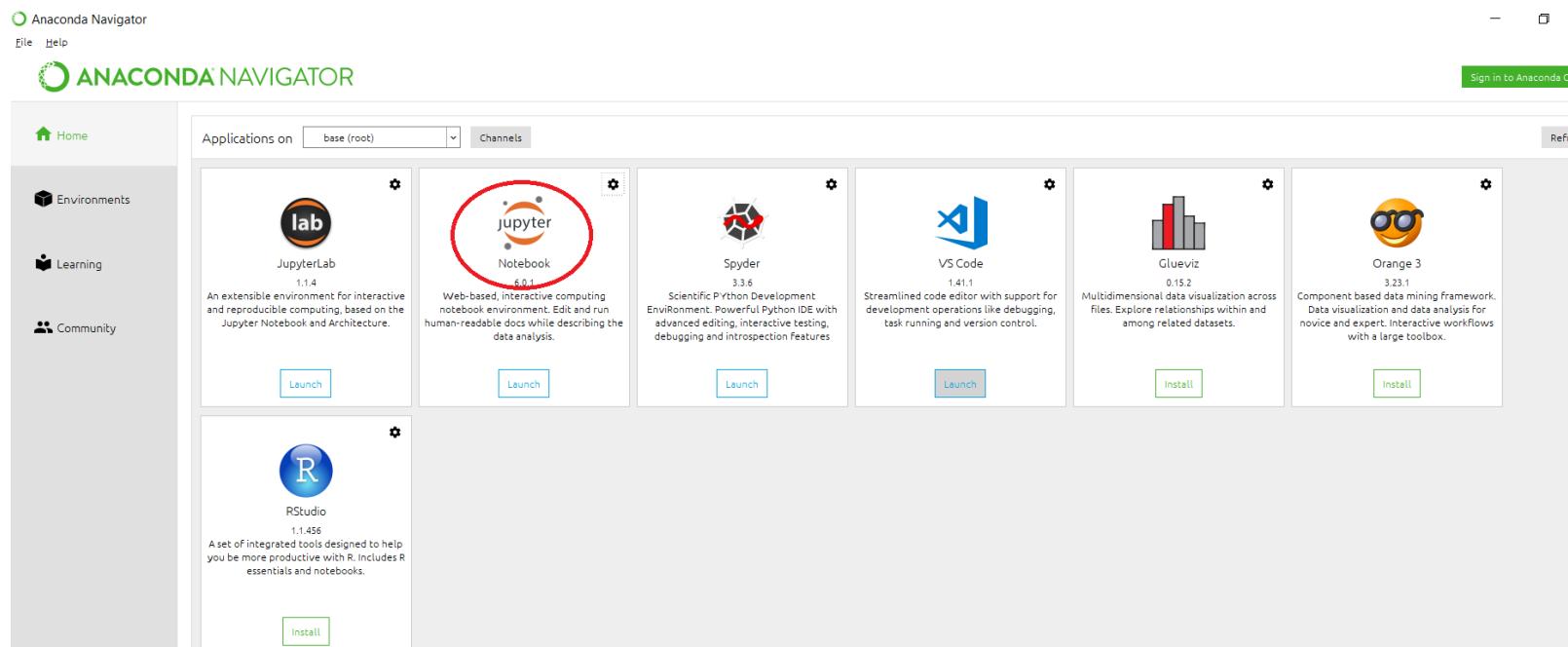
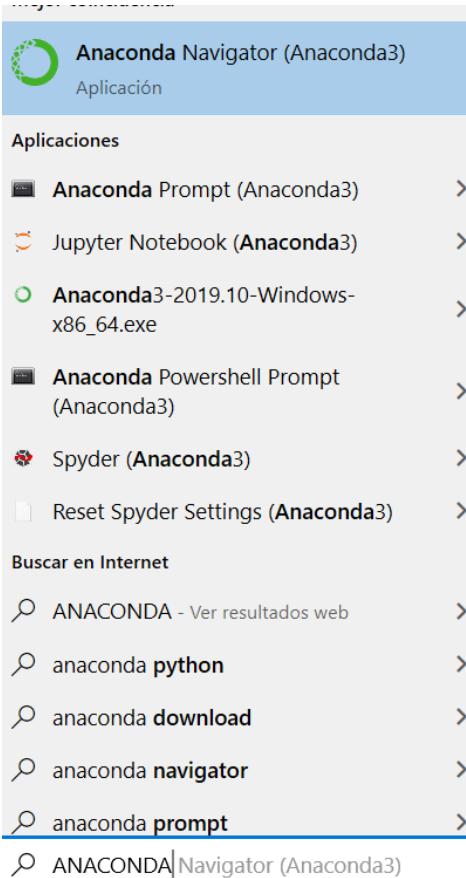
Python

ANACONDA



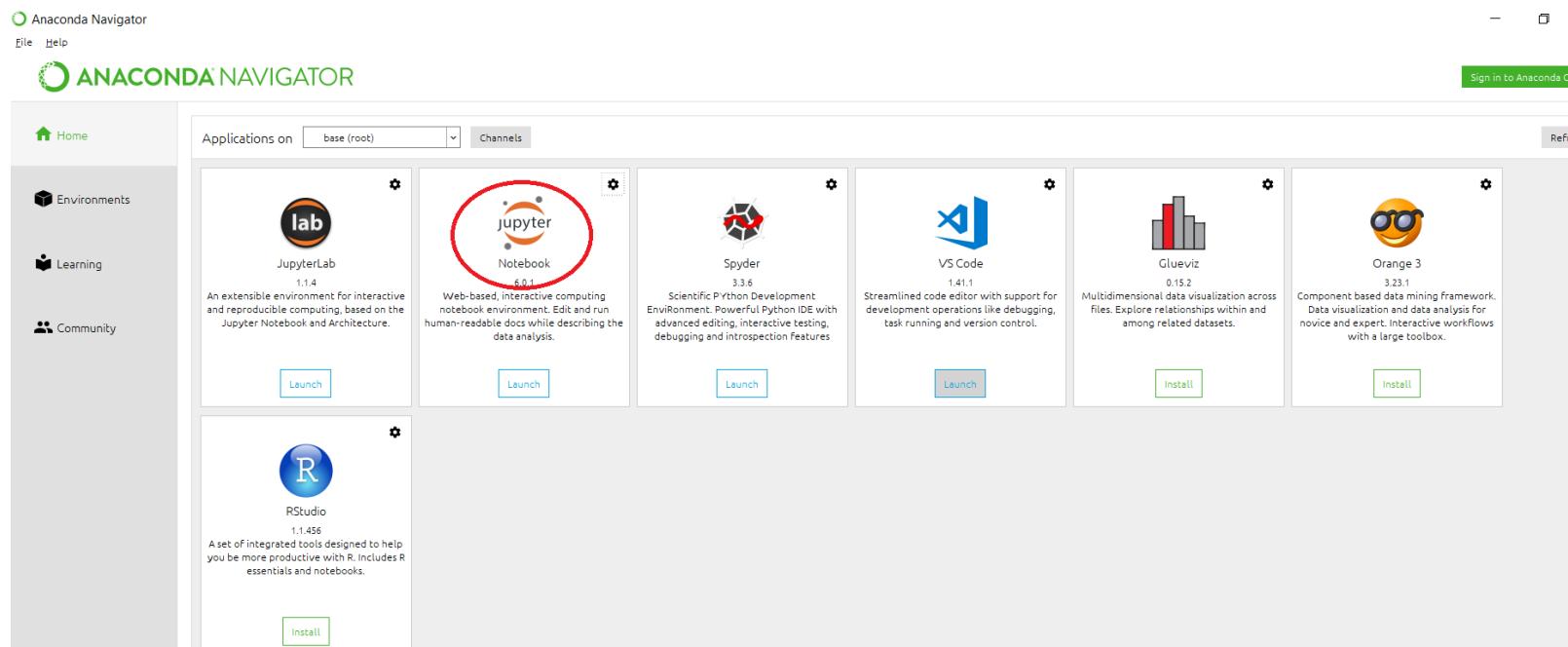
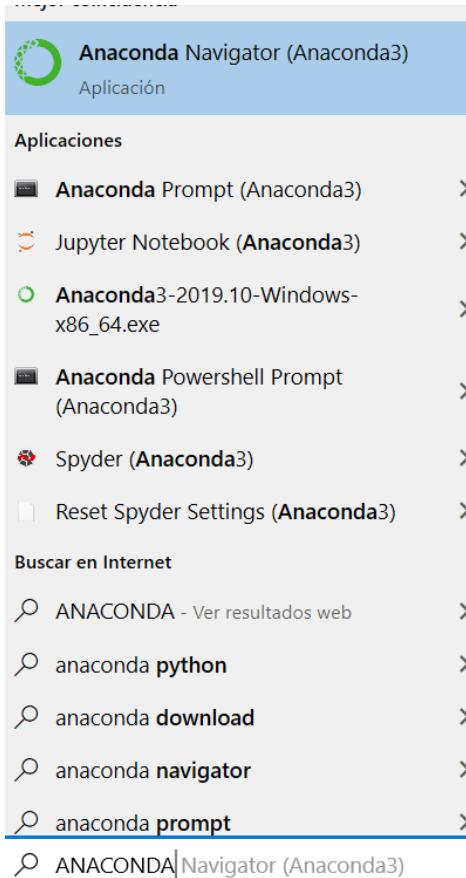
Python

ANACONDA



Python

ANACONDA



¡Muchas Gracias!



GABRIEL MARÍN DÍAZ
LCDO. CIENCIAS FÍSICAS UCM

www.linkedin.com/in/gabrielmarindiaz/