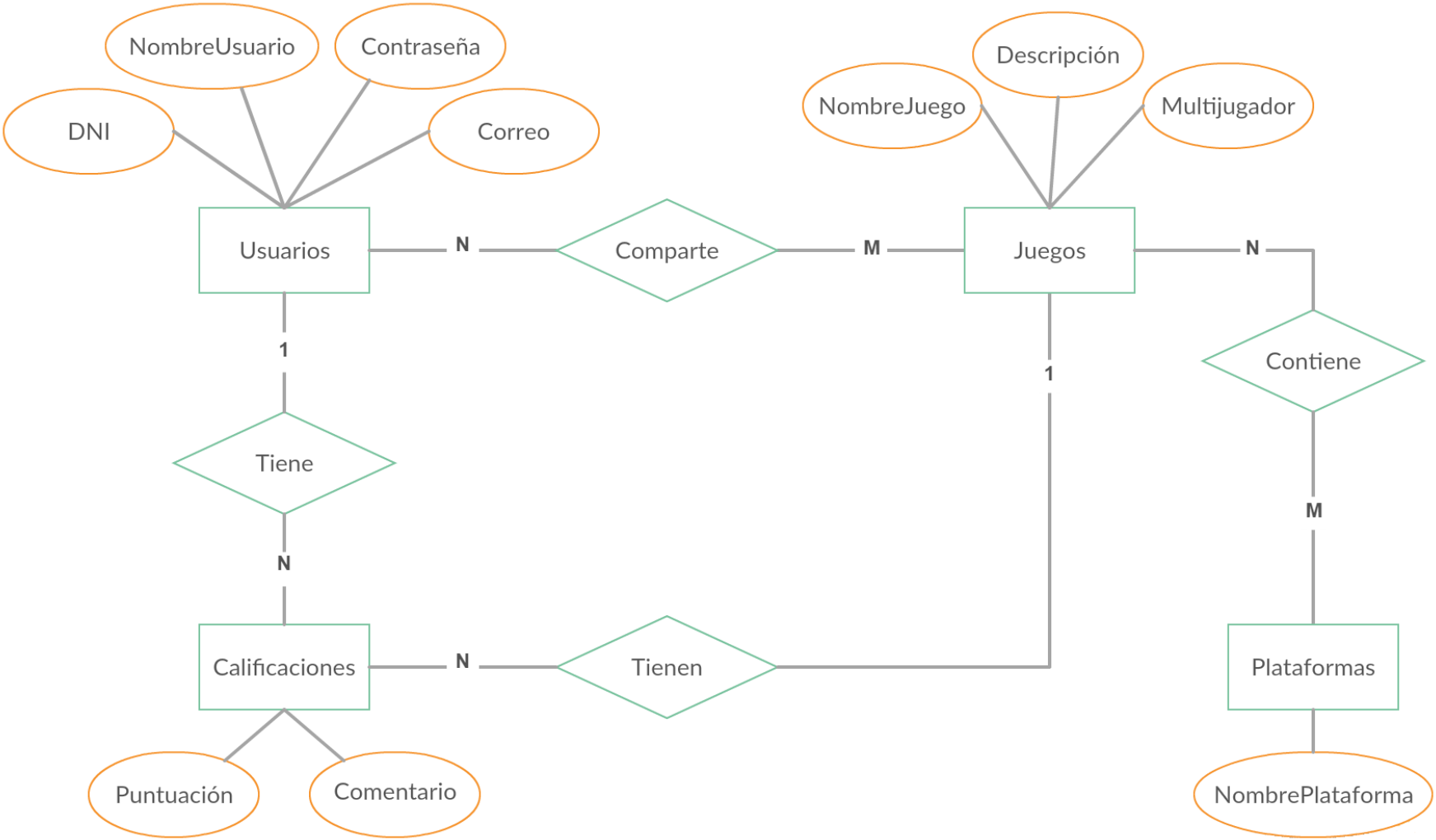
Planificación de base de datos

Proyecto: 4Games

Vitaly Sastre
Toni Rosselló
Tomeu Llompart
Miquel Ángel González

Índex

Modelo Entidad – Relación



Atributos y dominios

Usuarios	
Atributos	*Dominios*
DNI	Ax9 a Zx9
NombreUsuario	Ax45 a Zx45
Contraseña	Ax45 a Zx45
Correo	Ax100 a Zx100

Juegos	
Atributos	*Dominios*
NombreJuego	Ax45 a Zx45
Descripción	Ax100 a Zx100
Multijugador	Si o No

Calificaciones	
Atributos	*Dominios*
Puntuación	Entre 0 y 5
Comentario	Ax100 a Zx100

Plataformas	
Atributos	*Dominios*
NombrePlataforma	Ax45 a Zx45

Súper llaves

Definición

La súper llave es un conjunto de uno o más atributos que, juntos, permiten identificar en forma única a una entidad dentro del conjunto de entidades.

Entidad Usuarios:

- DNI
- DNI + NombreUsuario
- DNI + NombreUsuario + Contraseña
- DNI + NombreUsuario + Contraseña + Correo

Entidad Juegos:

- NombreJuego
- NombreJuego + Descripción
- NombreJuego + Descripción + Multijugador

Entidad Calificaciones:

- No existe un conjunto de atributos suficientes para formar una llave única. Por lo tanto esta entidad se define como **Entidad débil** y se le asignará un discriminador llamado **IdCalificación**.

Entidad Plataformas:

- No existe un conjunto de atributos suficientes para formar una llave única. Por lo tanto esta entidad se define como **Entidad débil** y se le asignará un discriminador llamado **IdPlataforma**.

Llaves candidato

Definición

Lo que se busca es la súper llave más pequeña posible. Estas súper llaves mínimas se denominan llaves candidato.

Entidad Usuarios:

- DNI.

Entidad Juegos:

- NombreJuego.

Entidad Calificaciones:

- IdCalificación.

Entidad Plataformas:

- IdPlataforma.

Llaves Primarias

Definición

Se utilizará el término llave primaria para referirse a la llave candidato que elija el diseñador de la base de datos como la forma principal de identificar a las entidades dentro de un conjunto de éstas.

Entidad Usuarios:

- DNI.

Entidad Juegos:

- NombreJuego.

Entidad Calificaciones:

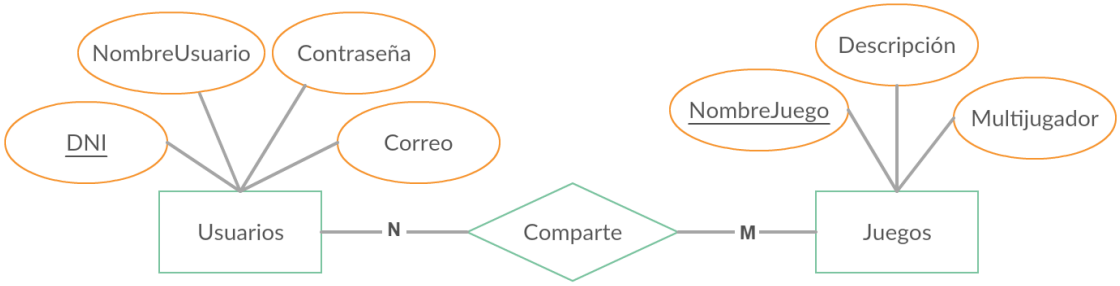
- IdCalificación.

Entidad Plataformas:

- IdPlataforma.

Modelo Relacional

Relación entre Usuarios y Juegos (N:M)



Usuarios = (DNI, NombreUsuario, Contraseña, Correo)

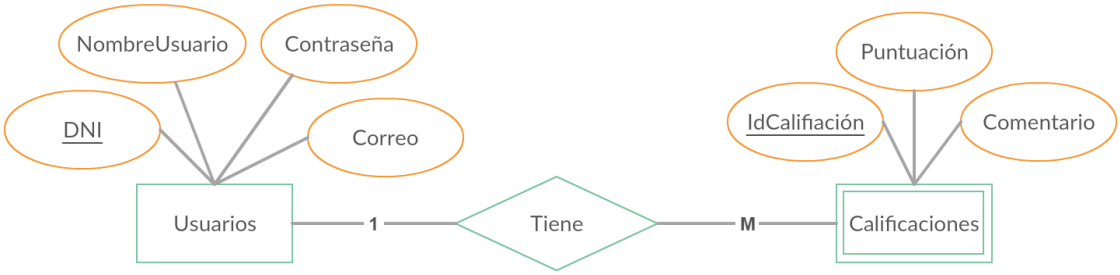
Juegos = (NombreJuego, Descripción, Multijugador)

Comparte = (DNI, NombreJuego)

========== OPTIMIZACIÓN ==========

No tiene optimización ya que las dos llaves primarias se comparten. Por lo tanto se creará una nueva entidad con atributos de las llaves primarias de las otras dos entidades.

Relación entre Usuarios y Calificaciones (1:M)



Usuarios = (DNI, NombreUsuario, Contraseña, Correo)

Calificaciones = (IdCalificación, Puntuación)

Tiene = (DNI, IdCalificación)

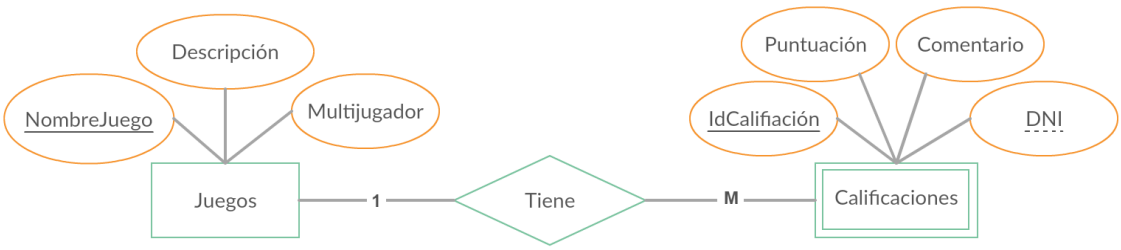
========== OPTIMIZACIÓN ==========

Usuarios = (DNI, NombreUsuario, Contraseña, Correo)

Calificaciones = (IdCalificación, Puntuación, DNI)

~~**Tiene** = (DNI, IdCalificación)~~

Relación entre Juegos y Calificaciones (1:M)



Juegos = (NombreJuego, Descripción, Multijugador)

Calificaciones = (IdCalificación, Puntuación, DNI)

Tienen = (NombreJuego, IdCalificación)

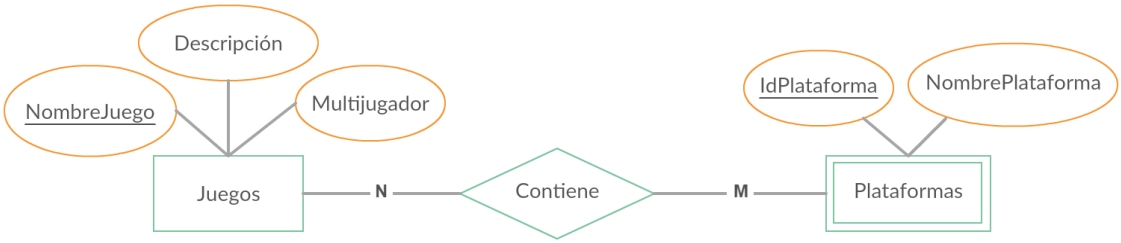
========== OPTIMIZACIÓN ==========

Juegos = (NombreJuego,Descripción, Multijugador)

Calificaciones = (IdCalificación, Puntuación, DNI, NombreJuego)

~~**Tienen** = (NombreJuego, IdCalificación)~~

Relación entre Juegos y Plataformas (N:M)



Juegos = (NombreJuego,Descripción, Multijugador)

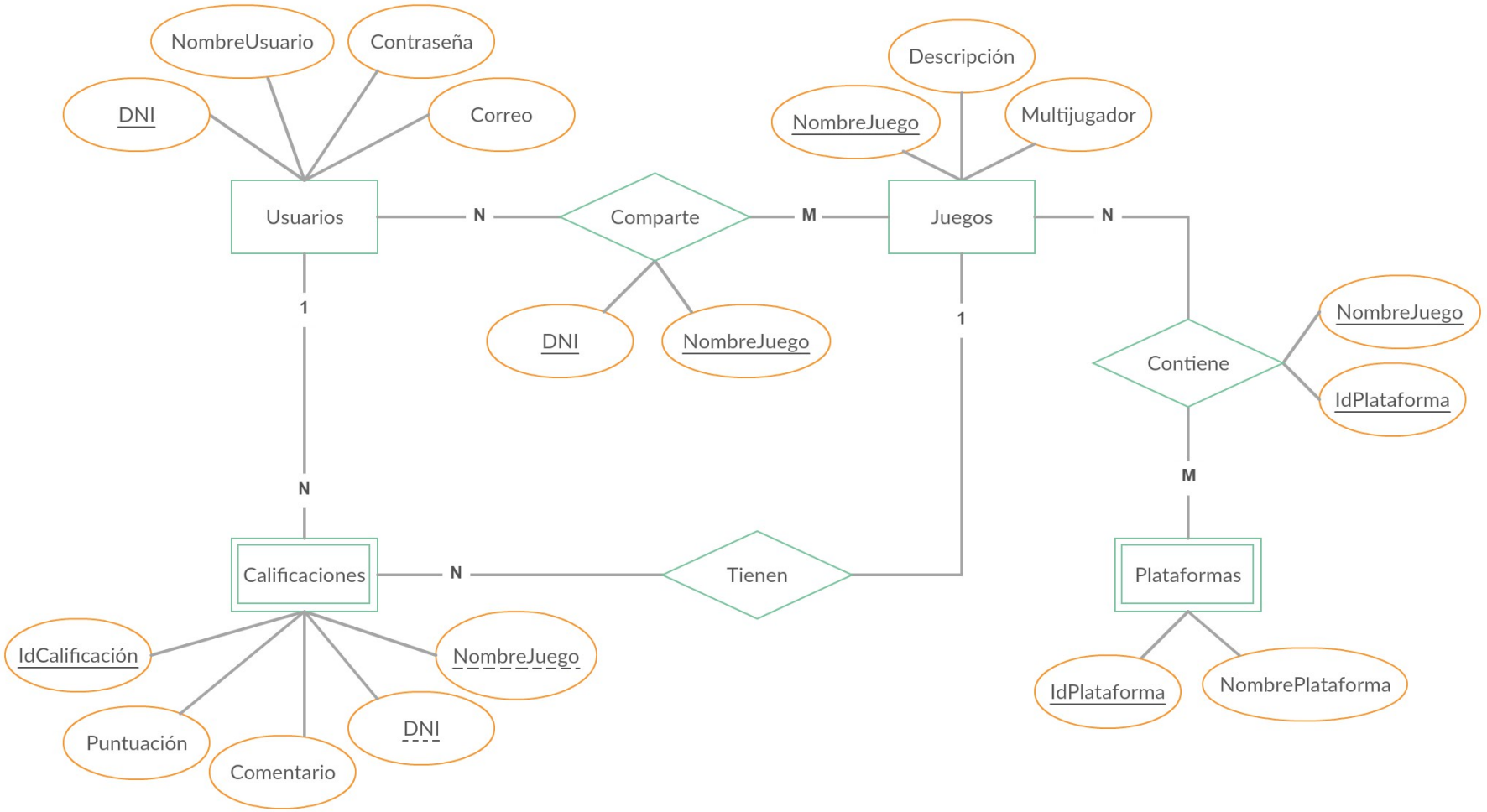
Plataformas = (IdPlataforma, NombrePlataforma)

Contienen = (NombreJuego, IdPlataforma)

========== OPTIMIZACIÓN ==========

No tiene optimización ya que las dos llaves primarias se comparten. Por lo tanto se creará una nueva entidad con atributos de las llaves primarias de las otras dos entidades.

Modelo Entidad - Relación optimizado



Reglas de integridad referencial

Regla de integridad de entidades

Entidad Usuarios:

- Su llave primaria es **DNI** y no puede tener valor nulo y tampoco puede repetirse.

Entidad Juegos:

- Su llave primaria es **NombreJuego** y no puede tener valor nulo y tampoco puede repetirse.

Entidad Calificaciones:

- Su llave primaria es **IdCalificación** y no puede tener valor nulo y tampoco puede repetirse.

Entidad Plataformas:

- Su llave primaria es **IdPlataforma** y no puede tener valor nulo y tampoco puede repetirse.

Relación Comparte:

- Sus llaves primarias son **DNI** y **NombreJuego** y no pueden tener valor nulo y tampoco pueden repetirse.

Relación Contiene:

- Sus llaves primarias son **NombreJuego** e **IdPlataforma** y no pueden tener valor nulo y tampoco pueden repetirse.

Regla de integridad referencial

Entidad Calificaciones:

- Sus llaves extranjeras son **DNI** y **NombreJuego** y ambas son consistentes.

Reglas de integridad para llaves extranjeras

Entidad Calificaciones:

- Sus llaves extranjeras son **DNI y NombreJuego**.
    - DNI:
        - Sí puede ser nulo.
        - Si se intenta eliminar la llave primaria referenciada por una llave extranjera: **No hacer nada**.
        - Si se intenta modificar la llave primaria referenciada por una llave extranjera: **No hacer nada**.
    - NombreJuego:
        - No puede ser nulo.
        - Si se intenta eliminar la llave primaria referenciada por una llave extranjera: **Eliminar en cascada**.
        - Si se intenta modificar la llave primaria referenciada por una llave extranjera: **Modificar en cascada**.

Formas normales sobre base de datos relacionales

Dependencia funcional o dependencia funcional plena

Concepto

Sea R una relación, y X e Y dos atributos de R.

Se dice que X determina funcionalmente a Y o Y depende funcionalmente de X si y solo si: Para cada valor de X hay una y solo una imagen de Y.

Usuarios = (DNI, NombreUsuario, Contraseña, Correo)

Juegos = (NombreJuego, Descripción, Multijugador)

Calificaciones = (IdCalificación, Puntuación, DNI, NombreJuego)

Plataformas = (IdPlataforma, NombrePlataforma)

Primera forma normal (1FN)

Todas las entidades sí están en 1FN por que todos los dominios sobre los que están definidos los atributos son atómicos.

Segunda forma normal (2FN)

Todas las entidades sí están en 2FN por que cumplen la norma de 1FN y además todo atributo que no es llave primaria depende funcionalmente de toda la llave primaria.

Tercera forma normal (3FN)

Todas las entidades sí están en 3FN por que cumple la norma de 2FN y además todo atributo que no es llave primaria no depende funcionalmente de ningún otro atributo que no sea llave primaria.

Traducción del modelo relacional de datos al modelo físico de datos

Reglas de traducción de relaciones a tablas

1ª Regla

Toda relación del modelo relacional se convierte en una tabla del modelo físico.

2ª Regla

Cada atributo de una relación del modelo relacional se convierte en un campo de una tabla del modelo físico.

3ª Regla

Cada dominio de los atributos de una relación del modelo relacional, definen el tipo de dato del campo de una tabla.

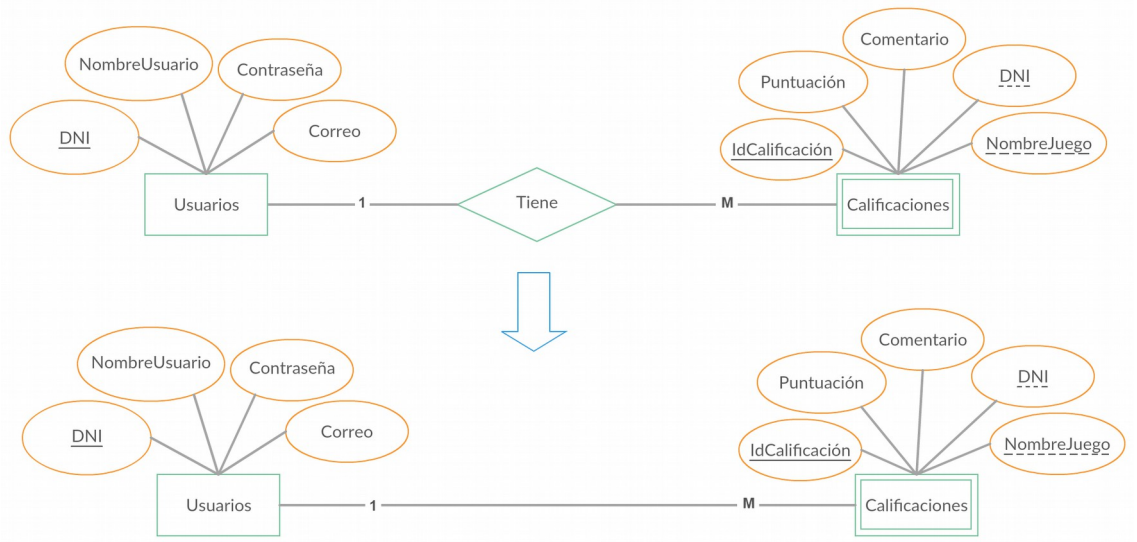
Reglas de cardinalidad

4ª Regla

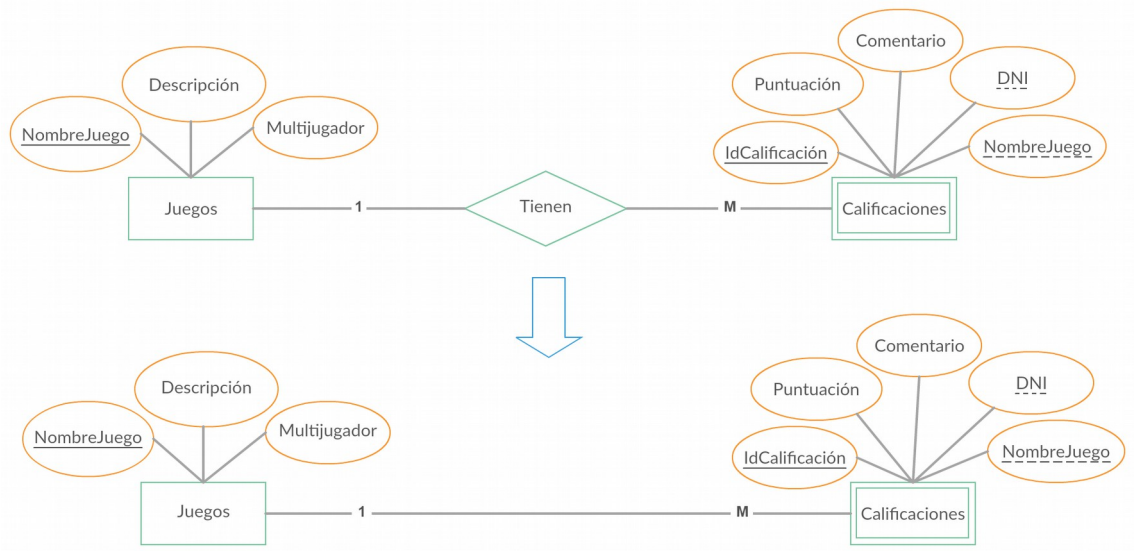
Las relaciones son de tipo 1:1, en este caso no existe ese tipo de relación.

5ª Regla

Relación entre Usuarios y Calificaciones (1:N)

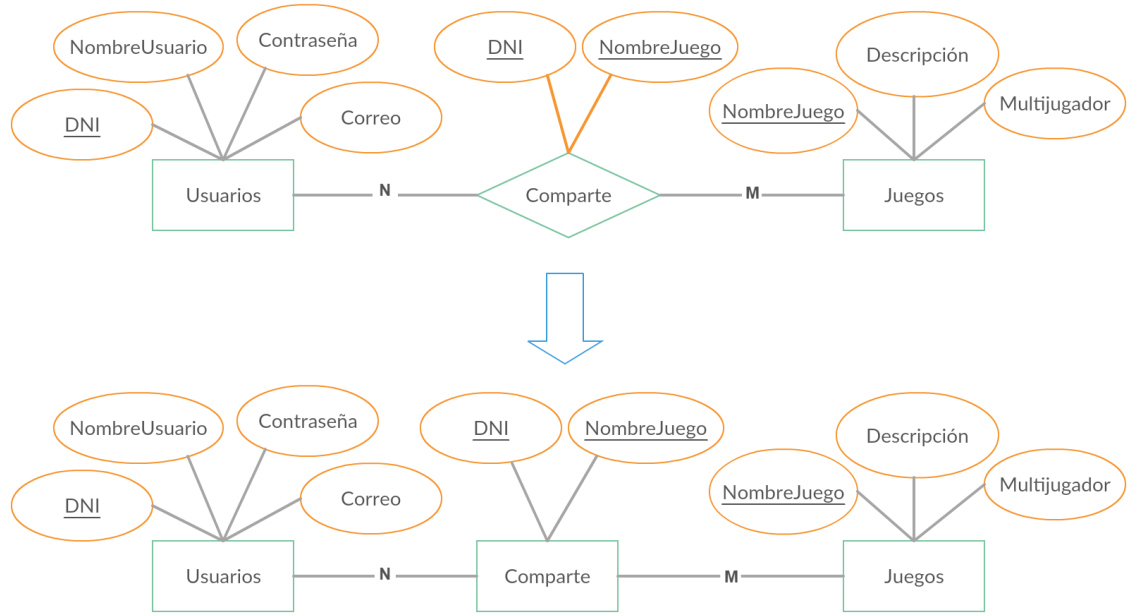


Relación entre Juegos y Calificaciones (1:N)

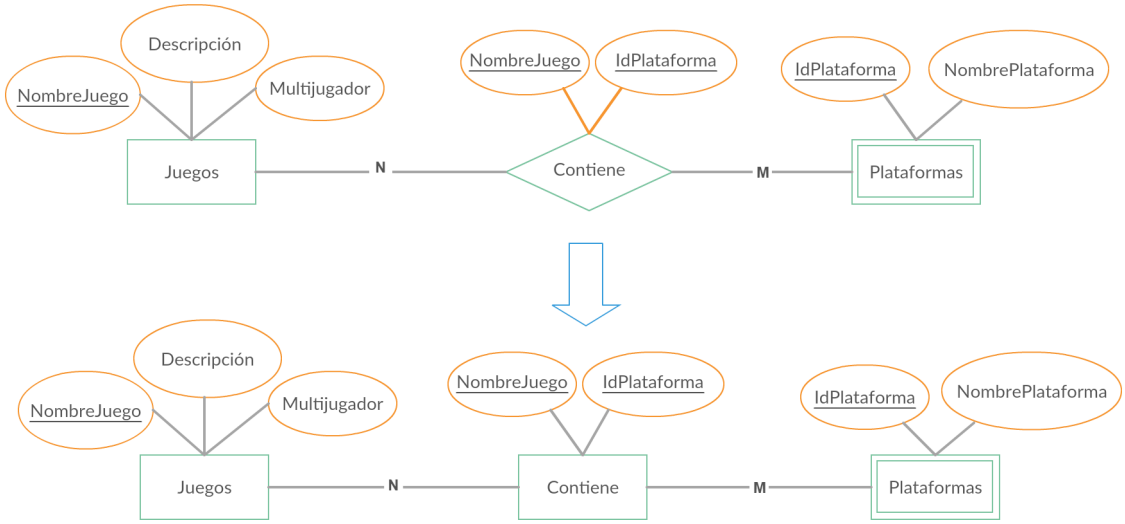


6ª Regla

Relación entre Usuarios y Juegos (N:M)



Relación entre Juegos y Plataformas (N:M)



Atributos y dominios de forma final

Usuarios	
Atributos	*Dominios*
DNI	Ax9 a Zx9
NombreUsuario	Ax45 a Zx45
Contraseña	Ax45 a Zx45
Correo	Ax100 a Zx100

Comparte	
Atributos	*Dominios*
DNI	Ax9 a Zx9
NombreJuego	0x9 a 9x9

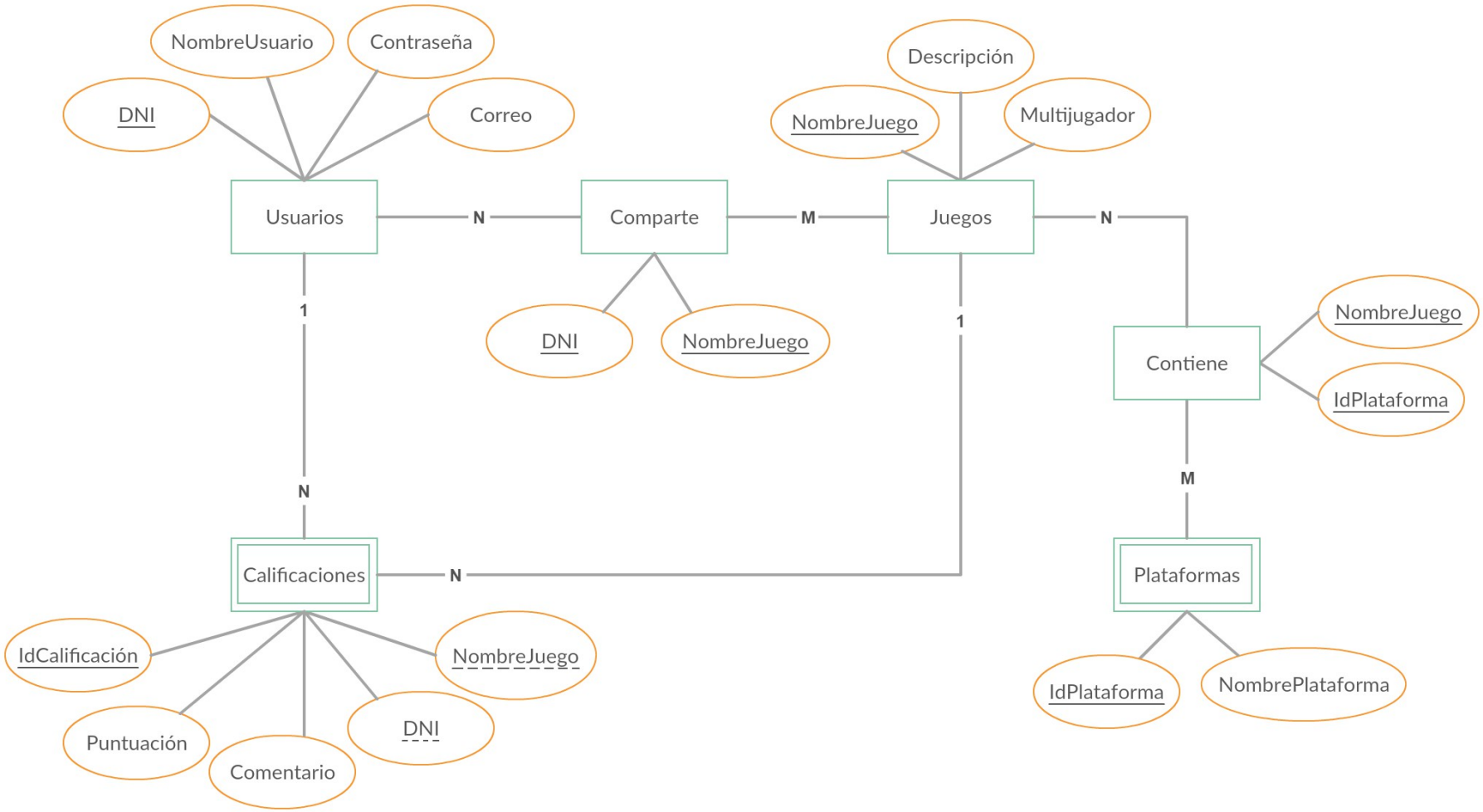
Juegos	
Atributos	*Dominios*
NombreJuego	Ax45 a Zx45
Descripción	Ax100 a Zx100
Multijugador	Si o No

Contiene	
Atributos	*Dominios*
NombreJuego	0x9 a 9x9
IdPlataforma	0x9 a 9x9

Plataformas	
Atributos	*Dominios*
IdPlataforma	0x9 a 9x9
NombrePlataforma	Ax45 a Zx45

Calificaciones	
Atributos	*Dominios*
IdCalificación	0x9 a 9x9
Puntuación	Entre 0 y 5
Comentario	Ax100 a Zx100
DNI	Ax9 a Zx9
NombreJuego	0x9 a 9x9

Forma final


NombreUsuario
Contraseña
DNI
Correo
Usuarios
N
Comparte
M
Juegos
N
Descripción
NombreJuego
Multijugador
DNI
NombreJuego
1
1
N
Contiene
NombreJuego
IdPlataforma
M
Calificaciones
N
Plataformas
IdCalificación
NombreJuego
Puntuación
DNI
Comentario
IdPlataforma
NombrePlataforma

Código consulta SQL creación de la base de datos

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

-- -----------------------------------------------------

-- Schema Juegos

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema Juegos

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `Juegos` DEFAULT CHARACTER SET utf8 ;

USE `Juegos` ;

-- -----------------------------------------------------

-- Table `Juegos`.`Usuarios`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `Juegos`.`Usuarios` ;

CREATE TABLE IF NOT EXISTS `Juegos`.`Usuarios` (

  `Dni` VARCHAR(9) NOT NULL,

  `NombreUsuario` VARCHAR(45) NOT NULL,

  `Password` VARCHAR(45) NOT NULL,

  `Email` VARCHAR(100) NOT NULL,

  PRIMARY KEY (`Dni`),

  UNIQUE INDEX `DniUsuarios_UNIQUE` (`Dni` ASC))

ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `Juegos`.`Juegos`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `Juegos`.`Juegos` ;
CREATE TABLE IF NOT EXISTS `Juegos`.`Juegos` (
  `NombreJuego` VARCHAR(45) NOT NULL,
  `DescripcionJuego` VARCHAR(100) NOT NULL,
  `Multijugador` TINYINT(1) NOT NULL,
  PRIMARY KEY (`NombreJuego`))
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `Juegos`.`Plataformas`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `Juegos`.`Plataformas` ;
CREATE TABLE IF NOT EXISTS `Juegos`.`Plataformas` (
  `IdPlataforma` MEDIUMINT UNSIGNED NOT NULL,
  `NombrePlataforma` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`IdPlataforma`))
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `Juegos`.`Juegos_has_Plataformas`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `Juegos`.`Juegos_has_Plataformas` ;
CREATE TABLE IF NOT EXISTS `Juegos`.`Juegos_has_Plataformas` (
  `Juegos_NombreJuego` VARCHAR(45) NOT NULL,
  `Plataformas_IdPlataforma` MEDIUMINT UNSIGNED NOT NULL,
  PRIMARY KEY (`Juegos_NombreJuego`, `Plataformas_IdPlataforma`),
  INDEX `fk_Juegos_has_Plataformas_Plataformas1_idx` (`Plataformas_IdPlataforma`
ASC),
  INDEX `fk_Juegos_has_Plataformas_Juegos1_idx` (`Juegos_NombreJuego` ASC),
  CONSTRAINT `fk_Juegos_has_Plataformas_Juegos1`
    FOREIGN KEY (`Juegos_NombreJuego`)
    REFERENCES `Juegos`.`Juegos` (`NombreJuego`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Juegos_has_Plataformas_Plataformas1`
    FOREIGN KEY (`Plataformas_IdPlataforma`)
    REFERENCES `Juegos`.`Plataformas` (`IdPlataforma`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `Juegos`.`Calificaciones`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `Juegos`.`Calificaciones` ;
CREATE TABLE IF NOT EXISTS `Juegos`.`Calificaciones` (
  `IdCalificacion` MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `Puntuacion` INT UNSIGNED NOT NULL,
  `Comentario` VARCHAR(100) NULL,
  `Juegos_NombreJuego` VARCHAR(45) NOT NULL,
  `Usuarios_Dni` VARCHAR(9) NOT NULL,
  PRIMARY KEY (`IdCalificacion`),
  INDEX `fk_Usuarios_has_Juegos1_Juegos1_idx` (`Juegos_NombreJuego` ASC),
  INDEX `fk_Usuarios_has_Juegos1_Usuarios1_idx` (`Usuarios_Dni` ASC),
  UNIQUE INDEX `Usuarios_DniUsuario_UNIQUE` (`Usuarios_Dni` ASC),
  CONSTRAINT `fk_Usuarios_has_Juegos1_Usuarios1`
    FOREIGN KEY (`Usuarios_Dni`)
    REFERENCES `Juegos`.`Usuarios` (`Dni`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Usuarios_has_Juegos1_Juegos1`
    FOREIGN KEY (`Juegos_NombreJuego`)
    REFERENCES `Juegos`.`Juegos` (`NombreJuego`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `Juegos`.`Usuarios_has_Juegos`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `Juegos`.`Usuarios_has_Juegos` ;
CREATE TABLE IF NOT EXISTS `Juegos`.`Usuarios_has_Juegos` (
  `Usuarios_Dni` VARCHAR(9) NOT NULL,
  `Juegos_NombreJuego` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`Usuarios_Dni`, `Juegos_NombreJuego`),
  INDEX `fk_Usuarios_has_Juegos_Juegos1_idx` (`Juegos_NombreJuego` ASC),
  INDEX `fk_Usuarios_has_Juegos_Usuarios1_idx` (`Usuarios_Dni` ASC),
  UNIQUE INDEX `Usuarios_DniUsuario_UNIQUE` (`Usuarios_Dni` ASC),
  CONSTRAINT `fk_Usuarios_has_Juegos_Usuarios1`
    FOREIGN KEY (`Usuarios_Dni`)
    REFERENCES `Juegos`.`Usuarios` (`Dni`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Usuarios_has_Juegos_Juegos1`
    FOREIGN KEY (`Juegos_NombreJuego`)
    REFERENCES `Juegos`.`Juegos` (`NombreJuego`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```