

Proyecto Integrado

*Realizado por:*

**Juan Ignacio Velasco Sánchez**

*Curso:*

**2016-2017**

Diciembre de 2016

Índice

[1. Presentación 2](#_Toc468147559)

[2. Base de datos 3](#_Toc468147560)

[2.1. Modelo Chen(A3) 4](#_Toc468147561)

[2.2. Modelo Martin 5](#_Toc468147562)

[2.3. Grafo Relacional 6](#_Toc468147563)

[2.4. DLL generado por Visio 7](#_Toc468147564)

[3. Aplicación Web 19](#_Toc468147565)

# 1. Presentación

Este proyecto consta de 3 partes principales:

Base de datos: en la cual se diseña la base de datos con la que va a funcionar la aplicación web, en nuestro caso, una aplicación web sobre valoraciones y comentarios sobre películas.

Aplicación Web: en esta parte se muestra la distribución de los archivos y el código fuente de la aplicación, y el manual para usuario normal y para administradores.

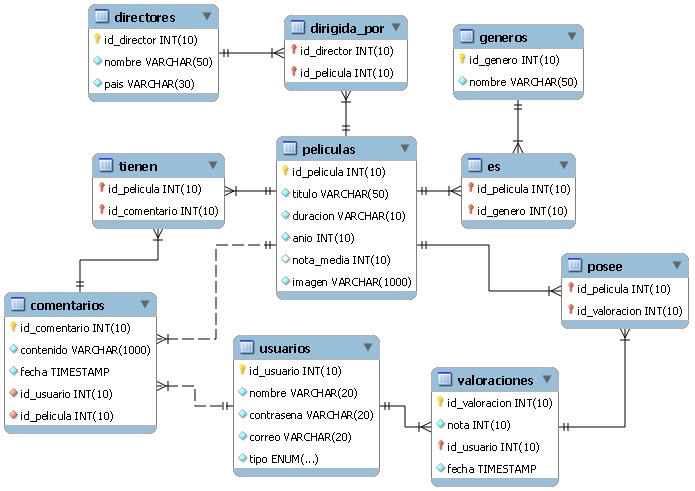
Redes: en esta parte se aborda una situación real, en la cual debemos realizar una instalación de red de una empresa desde los planos del edificio hasta las configuraciones de los dispositivos que intervienen en la red, subredes y otras características cumpliendo unos requisitos de seguridad y sin sobrepasar un presupuesto fijado como límite.

# 2. Base de datos

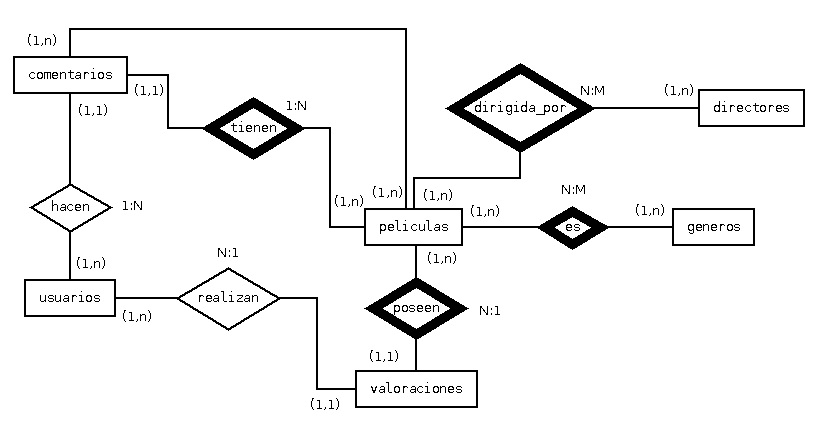
Para diseñar la BD de la aplicación web, se ha tenido en cuenta la mayor casuística posible dentro del contexto de la aplicación web, que se va a llamar ***FilmReview***,y que ha sido diseñada con el objetivo de que los usuarios puedan valorar películas, dejar valoraciones sobre las mismas, y ver en tiempo real notas medias de películas gracias a las valoraciones de otros usuarios. Veamos los modelos para entender la distribución de la información que se va a almacenar en la base de datos:

## 2.1. Modelo Chen(A3)

## 2.2. Modelo Martin



## 2.3. Grafo Relacional



## 2.4. DLL generado por Visio

-- This SQL DDL script was generated by Visio Enterprise (Release Date: LOCAL BUILD).

-- Driver Used : Visio Enterprise - ODBC Generic Driver Driver.

-- Document : c:\documents and settings\admin\escritorio\dll\_final.vsd.

-- Time Created: 27 de November de 2016 18:02.

-- Operation : From Visio Generate Wizard.

-- Connected data source : origen

-- Connected server : ASI

-- Connected database :

-- Create new table posee.

-- posee : Table of posee

-- id\_pelicula : id\_pelicula identifies posee

-- id\_valoracion : id\_valoracion partly identifies posee

create table posee (

id\_pelicula INTEGER not null constraint poseeid\_pelicula\_Chk check (id\_pelicula between 0 and 9999999999) ,

id\_valoracion INTEGER not null constraint poseeid\_valoracion\_Chk check (id\_valoracion between 0 and 9999999999) , constraint posee\_PK primary key (id\_pelicula, id\_valoracion) )

-- Create new table valoraciones.

-- valoraciones : Table of valoraciones

-- id\_valoracion : id\_valoracion identifies valoraciones

-- nota : nota is of valoraciones

-- id\_usuario : id\_usuario is of valoraciones

-- fecha : fecha is of valoraciones

create table valoraciones (

id\_valoracion INTEGER not null constraint valoracionesid\_valoracion\_Chk check (id\_valoracion between 0 and 9999999999) ,

nota INTEGER null constraint valoracionesnota\_Chk check (nota between 0 and 9999999999) ,

id\_usuario INTEGER null constraint valoracionesid\_usuario\_Chk check (id\_usuario between 0 and 9999999999) ,

fecha TIMESTAMP null, constraint valoraciones\_PK primary key (id\_valoracion) )

-- Create new table usuarios.

-- usuarios : Table of usuarios

-- id\_usuario : id\_usuario identifies usuarios

-- nombre : nombre is of usuarios

-- contrasena : contrasena is of usuarios

-- correo : correo is of usuarios

-- tipo : tipo is of usuarios

create table usuarios (

id\_usuario INTEGER not null constraint usuariosid\_usuario\_Chk check (id\_usuario between 0 and 9999999999) ,

nombre VARCHAR(20) null,

contrasena VARCHAR(20) null,

correo VARCHAR(20) null,

tipo VARCHAR(8) null, constraint usuarios\_PK primary key (id\_usuario) )

-- Create new table comentarios.

-- comentarios : Table of comentarios

-- id\_comentario : id\_comentario identifies comentarios

-- contenido : contenido is of comentarios

-- fecha : fecha is of comentarios

-- id\_usuario : id\_usuario is of comentarios

-- id\_pelicula : id\_pelicula is of comentarios

create table comentarios (

id\_comentario INTEGER not null constraint comentariosid\_comentario\_Chk check (id\_comentario between 0 and 9999999999) ,

contenido VARCHAR(250) null,

fecha TIMESTAMP null,

id\_usuario INTEGER null constraint comentariosid\_usuario\_Chk check (id\_usuario between 0 and 9999999999) ,

id\_pelicula INTEGER null constraint comentariosid\_pelicula\_Chk check (id\_pelicula between 0 and 9999999999) , constraint comentarios\_PK primary key (id\_comentario) )

-- Create new table generos.

-- generos : Table of generos

-- id\_genero : id\_genero identifies generos

-- nombre : nombre is of generos

create table generos (

id\_genero INTEGER not null constraint generosid\_genero\_Chk check (id\_genero between 0 and 9999999999) ,

nombre VARCHAR(50) null, constraint generos\_PK primary key (id\_genero) )

-- Create new table es.

-- es : Table of es

-- id\_genero : id\_genero identifies es

-- id\_pelicula : id\_pelicula partly identifies es

create table es (

id\_genero INTEGER not null constraint esid\_genero\_Chk check (id\_genero between 0 and 9999999999) ,

id\_pelicula INTEGER not null constraint esid\_pelicula\_Chk check (id\_pelicula between 0 and 9999999999) , constraint es\_PK primary key (id\_genero, id\_pelicula) )

-- Create new table tienen.

-- tienen : Table of tienen

-- id\_comentario : id\_comentario identifies tienen

-- id\_pelicula : id\_pelicula partly identifies tienen

create table tienen (

id\_comentario INTEGER not null constraint tienenid\_comentario\_Chk check (id\_comentario between 0 and 9999999999) ,

id\_pelicula INTEGER not null constraint tienenid\_pelicula\_Chk check (id\_pelicula between 0 and 9999999999) , constraint tienen\_PK primary key (id\_comentario, id\_pelicula) )

-- Create new table directores.

-- directores : Table of directores

-- id\_director : id\_director identifies directores

-- nombre : nombre is of directores

-- pais : pais is of directores

create table directores (

id\_director INTEGER not null constraint directoresid\_director\_Chk check (id\_director between 0 and 9999999999) ,

nombre VARCHAR(50) null,

pais VARCHAR(30) null, constraint directores\_PK primary key (id\_director) )

-- Create new table dirigida\_por.

-- dirigida\_por : Table of dirigida\_por

-- id\_pelicula : id\_pelicula partly identifies dirigida\_por

-- id\_director : id\_director identifies dirigida\_por

create table dirigida\_por (

id\_pelicula INTEGER not null constraint dirigida\_porid\_pelicula\_Chk check (id\_pelicula between 0 and 9999999999) ,

id\_director INTEGER not null constraint dirigida\_porid\_director\_Chk check (id\_director between 0 and 9999999999) , constraint dirigida\_por\_PK primary key (id\_director, id\_pelicula) )

-- Create new table peliculas.

-- peliculas : Table of peliculas

-- id\_pelicula : id\_pelicula identifies peliculas

-- titulo : titulo is of peliculas

-- duracion : duracion is of peliculas

-- anio : anio is of peliculas

-- nota\_media : nota\_media is of peliculas

-- imagen : imagen is of peliculas

-- id\_director : id\_director is of peliculas

create table peliculas (

id\_pelicula INTEGER not null constraint peliculasid\_pelicula\_Chk check (id\_pelicula between 0 and 9999999999) ,

titulo VARCHAR(50) not null,

duracion VARCHAR(10) not null,

anio INTEGER not null constraint peliculasanio\_Chk check (anio between 0 and 9999999999) ,

nota\_media INTEGER null constraint peliculasnota\_media\_Chk check (nota\_media between 0 and 9999999999) ,

imagen VARCHAR(250) not null,

id\_director INTEGER null constraint peliculasid\_director\_Chk check (id\_director between 0 and 9999999999) , constraint peliculas\_PK primary key (id\_pelicula) )

-- Add foreign key constraints to table posee.

alter table posee

add constraint peliculas\_posee\_FK1 foreign key (

id\_pelicula)

references peliculas (

id\_pelicula)

alter table posee

add constraint valoraciones\_posee\_FK1 foreign key (

id\_valoracion)

references valoraciones (

id\_valoracion)

-- Add foreign key constraints to table valoraciones.

alter table valoraciones

add constraint usuarios\_valoraciones\_FK1 foreign key (

id\_usuario)

references usuarios (

id\_usuario)

-- Add foreign key constraints to table comentarios.

alter table comentarios

add constraint peliculas\_comentarios\_FK1 foreign key (

id\_pelicula)

references peliculas (

id\_pelicula)

alter table comentarios

add constraint usuarios\_comentarios\_FK1 foreign key (

id\_usuario)

references usuarios (

id\_usuario)

-- Add foreign key constraints to table es.

alter table es

add constraint generos\_es\_FK1 foreign key (

id\_genero)

references generos (

id\_genero)

alter table es

add constraint peliculas\_es\_FK1 foreign key (

id\_pelicula)

references peliculas (

id\_pelicula)

-- Add foreign key constraints to table tienen.

alter table tienen

add constraint peliculas\_tienen\_FK1 foreign key (

id\_pelicula)

references peliculas (

id\_pelicula)

alter table tienen

add constraint comentarios\_tienen\_FK1 foreign key (

id\_comentario)

references comentarios (

id\_comentario)

-- Add foreign key constraints to table dirigida\_por.

alter table dirigida\_por

add constraint directores\_dirigida\_por\_FK1 foreign key (

id\_director)

references directores (

id\_director)

alter table dirigida\_por

add constraint peliculas\_dirigida\_por\_FK1 foreign key (

id\_pelicula)

references peliculas (

id\_pelicula)

-- Add foreign key constraints to table peliculas.

alter table peliculas

add constraint dirigida\_por\_peliculas\_FK1 foreign key (

id\_director,

id\_pelicula)

references dirigida\_por (

id\_director,

id\_pelicula)

-- This is the end of the Visio Enterprise generated SQL DDL script.

En resumen, se han contemplado casuísticas tales como por ejemplo que un usuario pueda dar una única valoración o nota por película, para que no sean alterables las notas medias por un mismo usuario, y que puedan comentar una película tantas veces como se desee. Los usuarios están clasificados según el campo “tipo” de la tabla usuarios, donde pueden ser usuario “estándar” o “admin”.

# 3. Aplicación Web

En esta parte, veremos la distribución de los archivos y carpetas de la aplicación web, proceso de instalación de la aplicación , librerías, funciones principales, interfaz principal y manual de funcionalidades de los usuarios estándar y los administradores.

Home

