

Universidad de Guanajuato

División de Ingenierías Campus Irapuato-Salamanca

Ingeniería en sistemas

Sistemas de información

Dr. Juan Carlos Gómez Carranza

*Práctica: Sistema de un cine*

Oscar Montes Camberos

5 de mayo del 2019

**INTRODUCCIÓN:**

A lo largo de esta práctica se diseñó e implementó un sistema de un cine, en el cual los usuarios pueden comprar boletos para las diferentes funciones, todas éstas creadas previamente por un administrador.

**DIAGRAMA MODULAR:**

*Plataformas*: Escritorio

*Ambiente*: Cine

*Actores*: Administrador y usuario normal

*Entradas*:

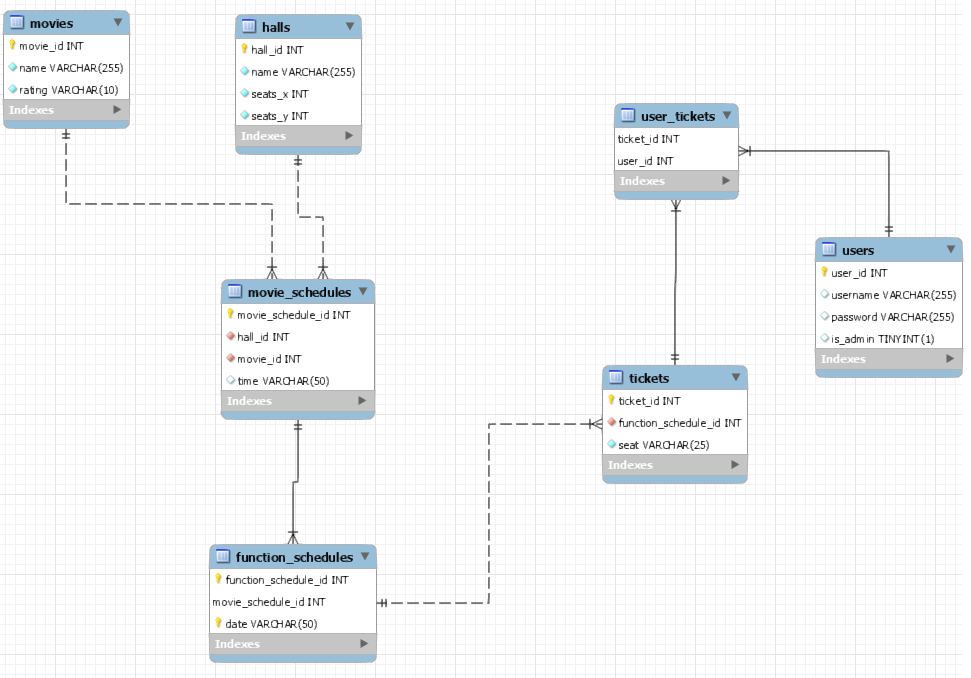
Información del usuario:

* ID de usuario
* Contraseña
* Información de salas, horarios, películas (en caso de ser administrador)

Información de acción:

* Teclado

*Diagrama*:

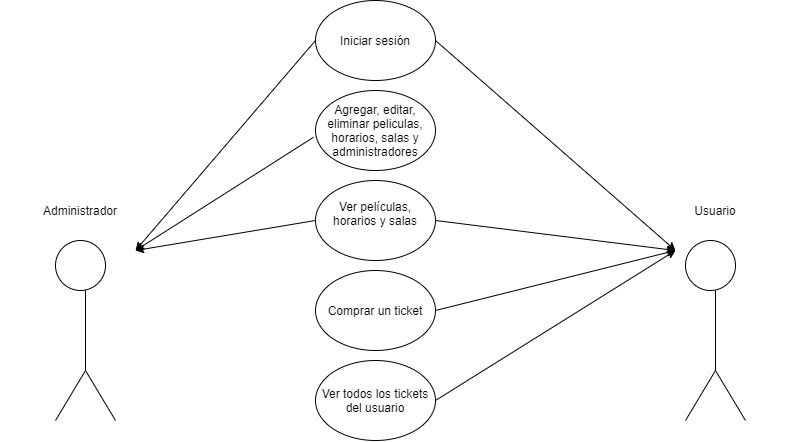


*Salidas*:

Usuario:

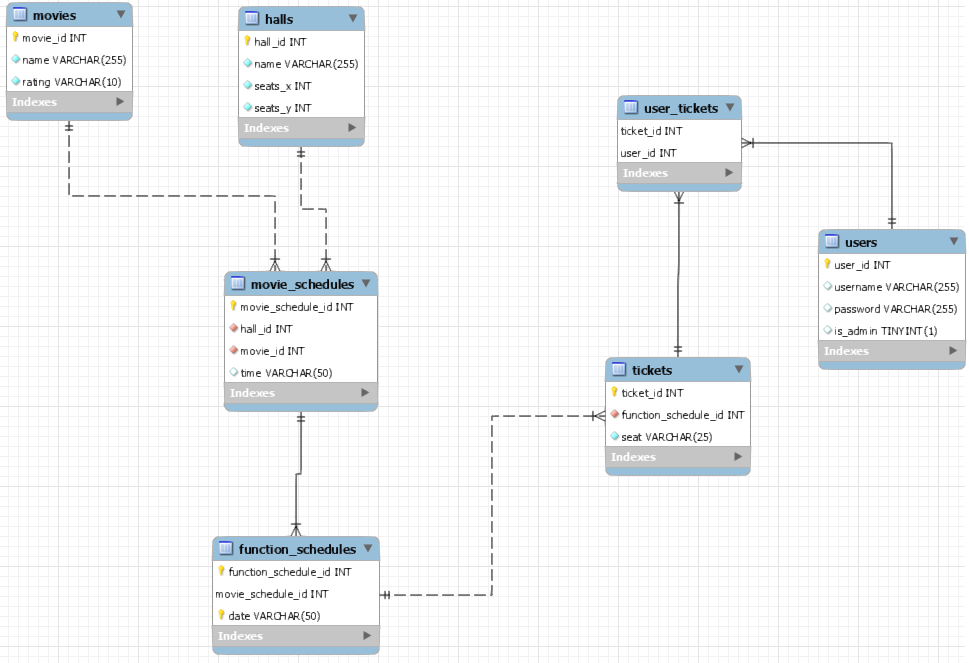
* Ticket de compra.

**DIAGRAMA CASOS DE USO:**



|  |  |
| --- | --- |
| Caso de uso | Ver todos los tickets del usuario |
| Actor | Usuario |
| Descripción | Permite visualizar los tickets comprados por un usuario en específico |
| Precondiciones | El usuario ya se ha iniciado sesión con un ID de usuario y contraseña |
| Postcondiciones | Se mostró con éxito los tickets comprados por el usuario |
| Suposiciones | El usuario tiene un ID de usuario y contraseña |
| Pasos a realizar: | |
| 1. El usuario solicita al sistema comenzar con el proceso de mostrar los tickets comprados por el usuario. 2. El sistema busca los tickets comprados en la base de datos según el ID de usuario y los muestra al cliente en la pantalla. | |

**DIAGRAMA RELACIONAL SQL:**



**SNIPPETS DE CÓDIGO IMPORTANTE:**

Se incluyen a continuación fragmentos del código importantes:

*Comprar un ticket:*

    def buy\_ticket\_menu(self):

        movies = *self*.model.get\_schedules()

*self*.view.show\_movies(movies)

*self*.view.msg('Introduce el numero de la pelicula para la que deseas comprar el boleto:')

        number\_selected = input()

        movie\_selected = movies[int(number\_selected)-1][0]

*self*.view.msg('Introduce el numero del horario en la que quieres reservar:')

        movie\_schedules = movies[int(number\_selected)-1][2]

*self*.view.show\_movie\_schedules(movie\_schedules)

        number\_selected = input()

        schedule\_selected = movie\_schedules.split(',')[int(number\_selected)-1]

*self*.view.msg('Introduce el numero de la fecha en la que quieres reservar:')

        next\_days = *self*.model.get\_next\_days()

*self*.view.show\_next\_days(next\_days)

        number\_selected = input()

        date\_selected = next\_days[int(number\_selected)-1]

*self*.view.msg('Introduce el asiento que quieres reservar, ejemplo: B6')

        schedule\_exists = *self*.model.schedule\_exists(date\_selected, schedule\_selected)

        occupied\_seats = ''

        if schedule\_exists:

            data = *self*.model.get\_occupied\_seats(movie\_selected, date\_selected, schedule\_selected)

            occupied\_seats = data[1].split(',')

        else:

            occupied\_seats = ['ZZZ',]

        seats = []

        hall\_capacity = *self*.model.get\_hall\_capacity(movie\_selected,schedule\_selected.strip())

        for x in range(hall\_capacity[0]):

            for y in range(hall\_capacity[1]):

                seats.append(f'{string.ascii\_lowercase[x].upper()}{y+1}')

        available\_seats = [b for b in seats if

            all(a not in b for a in occupied\_seats)]

*self*.view.msg(', '.join(available\_seats))

        seat\_selected = '0'

        while seat\_selected.upper() not in available\_seats:

            seat\_selected = input().upper()

            if seat\_selected == '0':

                seat\_selected = input().upper()

            elif seat\_selected.upper() not in available\_seats:

*self*.view.error('El asiento no está disponible, escoge otro:')

*self*.view.msg(', '.join(available\_seats))

        if not schedule\_exists:

            movie\_schedule\_id = hall\_capacity[2]

*self*.model.create\_function\_schedule(movie\_schedule\_id, date\_selected)

        function\_schedule\_id = *self*.model.get\_function\_schedule\_id(movie\_selected, date\_selected, schedule\_selected.strip())

*self*.model.create\_ticket(function\_schedule\_id, *self*.user\_id, seat\_selected)

*self*.view.show\_order\_details(movie\_selected, date\_selected, schedule\_selected, seat\_selected)

*self*.view.msg('¡Compraste el boleto de manera exitosa!, presiona cualquier tecla para continuar.')

        tmp = input()

*self*.main\_menu(*self*.is\_admin)

*Consulta SQL para obtener los boletos comprados por un cliente:*

-- Get user tickets

SELECT movies.name, function\_schedules.date, movie\_schedules.time, halls.name, GROUP\_CONCAT(tickets.seat SEPARATOR ", ") AS seats FROM user\_tickets

JOIN tickets ON user\_tickets.user\_id = tickets.user\_id

JOIN function\_schedules ON tickets.function\_schedule\_id = function\_schedules.function\_schedule\_id

JOIN movie\_schedules ON function\_schedules.movie\_schedule\_id = movie\_schedules.movie\_schedule\_id

JOIN movies ON movie\_schedules.movie\_id = movies.movie\_id

JOIN halls ON movie\_schedules.hall\_id = halls.hall\_id

AND user\_tickets.user\_id = USERID GROUP BY function\_schedules.function\_schedule\_id ORDER BY function\_schedules.date