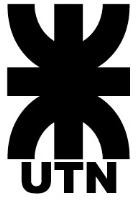


**UNIVERSIDAD TECNOLÓGICA NACIONAL FACULTAD REGIONAL CÓRDOBA**

**TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN ASIGNATURA: LABORATORIO DE COMPUTACIÓN II**

***Fecha de Entrega***: 05/10/2022

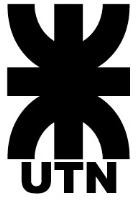


Contenido

[ENUNCIADO Nº 2: Venta de entradas en un complejo de cines 3](#_bookmark0)

[Diagrama de la Base de Datos 4](#_bookmark1)

Enunciado (Segunda Parte).…………………………………………….5



ENUNCIADO:

2.- Venta de entradas con butacas numeradas en un complejo de cines

Un conocido complejo de salas de exhibición de films de la Ciudad de Córdoba desea contar con un sistema que registre la compra de tickets por parte de los clientes. Para ello se debe diseñar una base de datos teniendo en cuenta la información relevada para tal fin:

Esta empresa posee varias salas de proyección de distintos tipos (común, vip, 3D, etc.), todas ellas con butacas numeradas. En el momento en que una persona se acerca a comprar entradas (a una ventanilla o por Internet), se emite un comprobante donde figuran los datos de la compra realizada (película, tipo de sala, función, cantidad de entradas, monto y forma de pago) y además un ticket por cada entrada donde debe constar la sala, la película, día, horario (función) y el nro. de butaca.

Una misma persona puede comprar, en un mismo momento varias entradas a distintas películas. Hay que tener en cuenta, que existen días y horarios promocionales, es decir con descuentos. Además, se pueden reservar entradas por internet la cual se confirma con el pago de estas. Esta reserva (si no se confirma) vence dos horas antes del inicio de función. Pueden basarse para el desarrollo de esta base de datos y la posterior programación de los informes una sala conocida por el grupo. Los reportes posibles para este sistema de información son, por ejemplo:

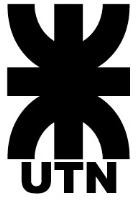
• Ingresos totales mensuales por día de proyección, o por temporada (vacaciones de invierno, de verano, feriados) o por bloques horarios.

• Tipo de público que asiste en distintos horario o temporadas, tipos de películas vistas, etc.

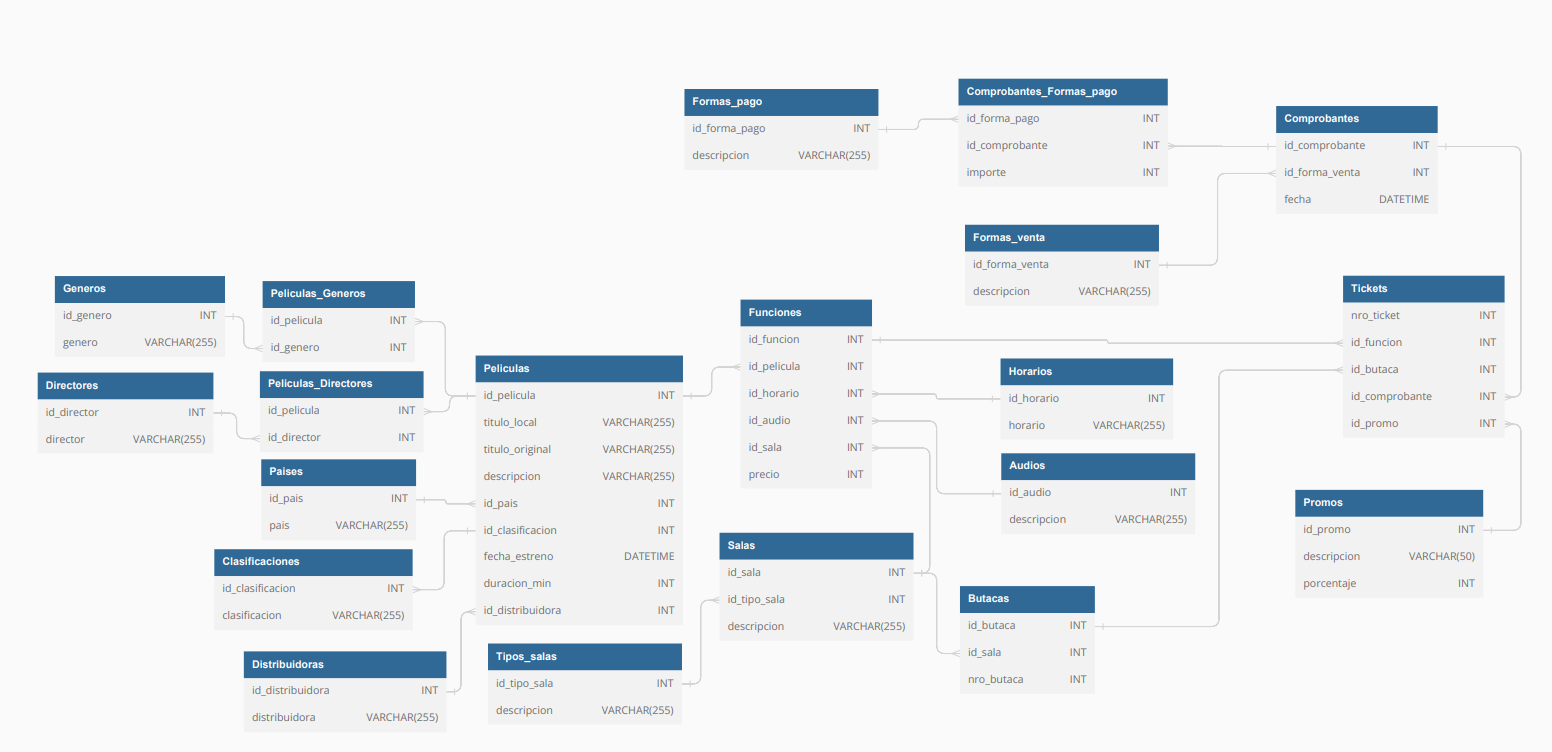
• Tipo de público (menores, mayores, jubilados) que nunca asistió a una determinada película o tipo de película (terror, acción, animadas, etc.)

• Horarios o películas que no tuvieron ningún asistente

• Películas más vistas.



# Diagrama de la Base de Datos



**Ilustración 1**: Elaboración Propia.

2da. Parte: enunciado de lo que se quiere mostrar en la consulta y código de la consulta que responda ese enunciado. Deberá entregarse junto con la primer parte en un único documento con un nombre apropiado, por ejemplo: 1WX\_GrupoX\_Parte2. Para entregar esta segunda parte, la primera debe estar ya aprobada. El nivel de complejidad de estas consultas debe estar de acuerdo con el grado de complicación de los ejercicios trabajados en clase, utilizando todos los conocimientos adquiridos en esta materia, donde existan composición de varias tablas, utilizando distintos tipos de test (distintas condiciones de búsquedas), consultas agrupadas, sumarias, subconsultas, vistas, procedimientos almacenados, funciones incorporadas y definidas por el usuario, triggers, tablas temporales y manejo de errores. Los resultados que se pretenden obtener de estas consultas pueden ser propuestos por el docente del curso o por los integrantes del curso. En este último caso, lo que se pretende mostrar, debe ser de utilidad para un supuesto usuario del Laboratorio de Computación II TPI Pág. 4 sistema (empleado, gerente, encargado o decisor de la entidad para la cual se diseña la propuesta).

--Mostrar la cantidad de Tikets que se vendieron el año pasado en el genero de Accion

SELECT COUNT(nro\_ticket) 'Cantidad de Entradas Vendidas'

FROM Tickets t join Funciones f on t.id\_funcion = f.id\_funcion

join Peliculas p on f.id\_pelicula = p.id\_pelicula

join Peliculas\_Generos pg on pg.id\_pelicula = p.id\_pelicula

join Generos g on g.id\_genero = pg.id\_genero

WHERE nro\_ticket in (SELECT nro\_ticket

FROM Comprobantes c join tickets t1 on c.id\_comprobante = t1.id\_comprobante

WHERE year(fecha) = year(GETDATE())-1 ) and genero like 'Accion%'

--- TOP 5 de las funciones mas vendidas del mes pasado

SELECT TOP 5 count(nro\_ticket)'Funciones Vendidas',f.id\_funcion, f.id\_pelicula 'Pelicula'

FROM Tickets t join Funciones f on t.id\_funcion = f.id\_funcion

join Peliculas p on f.id\_pelicula = p.id\_pelicula

WHERE f.id\_funcion in (SELECT id\_funcion

FROM Comprobantes c join Tickets t1 on c.id\_comprobante = t1.id\_comprobante

WHERE datediff(month,fecha,GETDATE())=1)

GROUP BY f.id\_funcion, f.id\_pelicula

ORDER BY 1 desc

--Monto total facturado Mensualmente

select sum(f.precio\*p.porcentaje/100),month(fecha),year(fecha)

from tickets t

join Comprobantes c on c.id\_comprobante=t.id\_comprobante

join promos p on p.id\_promo = t.id\_promo

join funciones f on f.id\_funcion = t.id\_funcion

group by month(fecha),year(fecha)

--Monto total facturado Anualmente

select sum(f.precio\*p.porcentaje/100),year(fecha)

from tickets t

join Comprobantes c on c.id\_comprobante=t.id\_comprobante

join promos p on p.id\_promo = t.id\_promo

join funciones f on f.id\_funcion = t.id\_funcion

group by year(fecha)

--Cantidad total de entradas vendidas este año

select count(nro\_ticket)'Cantidad de tickets'

from tickets t

join Comprobantes c on c.id\_comprobante=t.id\_comprobante

where DATEDIFF(year, year(fecha),year(getdate())) = 0

---Cantidad de entradas vendidas por sala este año

select count(nro\_ticket)'Cantidad de tickets', s.id\_sala

from tickets t

join Comprobantes c on c.id\_comprobante=t.id\_comprobante

join Butacas b on b.id\_butaca = t.id\_butaca

join Salas s on s.id\_sala = b.id\_sala

where DATEDIFF(year, year(fecha),year(getdate())) = 0

group by s.id\_sala

---monto total facturado por sala y por mes

select sum(f.precio\*p.porcentaje/100) 'Monto total facturado por sala',s.id\_sala,month(fecha) 'Mes',year(fecha)'Año'

from tickets t

join Comprobantes c on c.id\_comprobante=t.id\_comprobante

join promos p on p.id\_promo = t.id\_promo

join funciones f on f.id\_funcion = t.id\_funcion

join Salas s on s.id\_sala = f.id\_sala

group by s.id\_sala,month(fecha),year(fecha)

--PROCEDIMIENTO ALMACENADO para ingresar tickets

CREATE PROC SP\_INSERTAR\_TICKETS

@id\_funcion int,

@id\_butaca int,

@id\_comprobante int,

@id\_promo int,

@nro\_ticket int output

AS

BEGIN

INSERT INTO Tickets (id\_funcion,id\_butaca,id\_comprobante,id\_promo)

values (@id\_funcion,@id\_butaca,@id\_comprobante,@id\_promo)

SET @nro\_ticket = SCOPE\_IDENTITY();

END

--PROCEDIMIENTO ALMACENADO para consultar la funcion

create proc sp\_CONSULTAR\_FUNCION

@id\_funcion int

AS

BEGIN

select b.id\_butaca,b.nro\_butaca, b.id\_sala,f.id\_funcion,t.nro\_ticket,t.id\_butaca

from Butacas b

join Funciones f on f.id\_sala=b.id\_sala

left join tickets t on b.id\_butaca=t.id\_butaca

where f.id\_funcion=@id\_funcion

END

--vista del ticket y su precio final

create view Ticket\_Precio as

select t.nro\_ticket,t.id\_funcion,t.id\_butaca,t.id\_comprobante,t.id\_promo, f.precio\*p.porcentaje/100 'Precio'

from tickets t

join Promos p on t.id\_promo=p.id\_promo

join Funciones f on t.id\_funcion=f.id\_funcion;

--vista del comprobante y su precio final

create view Comprobante\_Precio as

select c.id\_comprobante,c.fecha,c.id\_forma\_venta,sum(f.precio\*p.porcentaje/100) 'Precio'

from tickets t

join Comprobantes c on t.id\_comprobante=c.id\_comprobante

join Promos p on t.id\_promo=p.id\_promo

join Funciones f on t.id\_funcion=f.id\_funcion

group by c.id\_comprobante,c.fecha,c.id\_forma\_venta

