Universidad de Costa Rica

Sede del Pacífico

Arnoldo Ferreto Segura

Carrera

Informática y Tecnología Multimedia

Curso

Manejo de Bases de Datos

Profesor

Melber Dalorso Cruz

Estudiantes

Jorge Ignacio Elizondo Alvarado

B42338

Leandro Bello Delgado

B40948

Jeannette Vargas Varela

B47443

Marcos Molina López

B03960

Período

II Ciclo 2017

Objetivo general

Diseñar una base de datos relacional mediante el análisis de las necesidades de la empresa Cinemas de Costa Rica para la implementación de un modelo ajustado a su lógica de negocio seguro y eficaz.

Objetivos específicos

Construir la base de datos del negocio para almacenar la información de transacciones mediante el estudio del funcionamiento de las sucursales.

Agilizar los procesos de transacciones de información mediante la implementación de una base de datos relacional.

Requerimientos como administrador de la base de datos para Cinemas de Costa Rica

Modelo de negocio

Reglas del negocio

Se desea diseñar una Base de Datos para llevar un control de los Cines de Costa Rica. Los datos significativos a tener en cuenta son:

La entidad producto contendrá los atributos ID producto, el nombre del producto, la descripción y el precio.

La tabla detalle_facturaCocina, tendrá un Fac_id_FacturaCocina, una factura de la cocina, producto, cantidad,precio, subtotal, impuesto/ventas,total y el total en dólares.

La entidad cocina tendrá como identificador a un id cocina,id empleado y una especialidad.

La entidad clientes contará con un id clientes, la cédula,nombre, y apellido1 y apellido2. La entidad factura boletería implementará los atributos, id factura boletería, el id de la caja, el id del cliente, la fecha y el tipo de pago.

La entidad caja tendrá el id caja, el id del empleado, el usuario y el password. La entidad empleado utilizará el id empleado,la cédula, el nombre, apellido1 , apellido2, direccion y contacto.

La entidad salario utilizara los atributos id salario,id empleado,hora normal,hora extra y precio de la hora.

La boletería salario tiene un id boleta de salario, un id salario,un monto y la fecha hora. La entidad salon tendrá como atributos el id salon y el id del empleado.

El detalle de la factura de la boletería tendrá su respectivo id detalle factura, el id factura de la boleta, el id de la película, la cantidad, el precio, el subtotal, el impuesto de ventas, el total y el total en dólares.

La entidad película con sus atributos id de la película, el nombre de la película,la fecha y la censura.

La entidad sala y sus atributos id sala,id butaca,id de la película,y la capacidad. La entidad boleto película son los atributos id boleto,id película e id de la sala. La entidad butaca tendrá de identificadores al ide butaca y el estado.

La entidad bitácora contará con un id bitácora,un usuario, un servidor, datos, fecha y transacción.

Consideraciones de Diseño

Un Empleado se identifica por su ID,cédula, nombre, los dos apellidos, dirección además de su contacto. Cada empleado recibe un salario según su id de empleado, las horas de trabajo ,horas extra y el valor de la hora laborada.

La entrega del salario a los empleado genera una boleta que contiene el identificador de la boleta, el identificador del salario, el monto y la fecha de pago.

Hay varios puestos de trabajos, como el que maneja la caja, los que están en el salón , los que están ubicados en la cocina, cada uno se identifica por su id y nombre del puesto. Los empleados de caja poseen un password con su respectivos usuario, en la cocina tienen roles según su especialidad.

Las salas de cada sucursal tiene sus propios atributos como el ID de la sala, su capacidad de público, otro atributo que necesita es el ID de la película y la capacidad de la sala.

Películas, necesita de un identificador osea un ID, el nombre de la película,la fecha de estreno de la película, la fecha de estreno de la misma además un identificador de categoría de la censura. La película va estar ligada con la sala, para cada sala va haber un boleto con los identificadores ID del boleto, el ID de la Factura y el ID de la sala.

Cada Película genera un facura diferente con los atributos ID del detalle de la factura,ID de la factura de la boleta, el ID de la Película, la cantidad de boletos comprados, el precio, el subtotal, el impuesto de la compra, el total y el total en dólares.

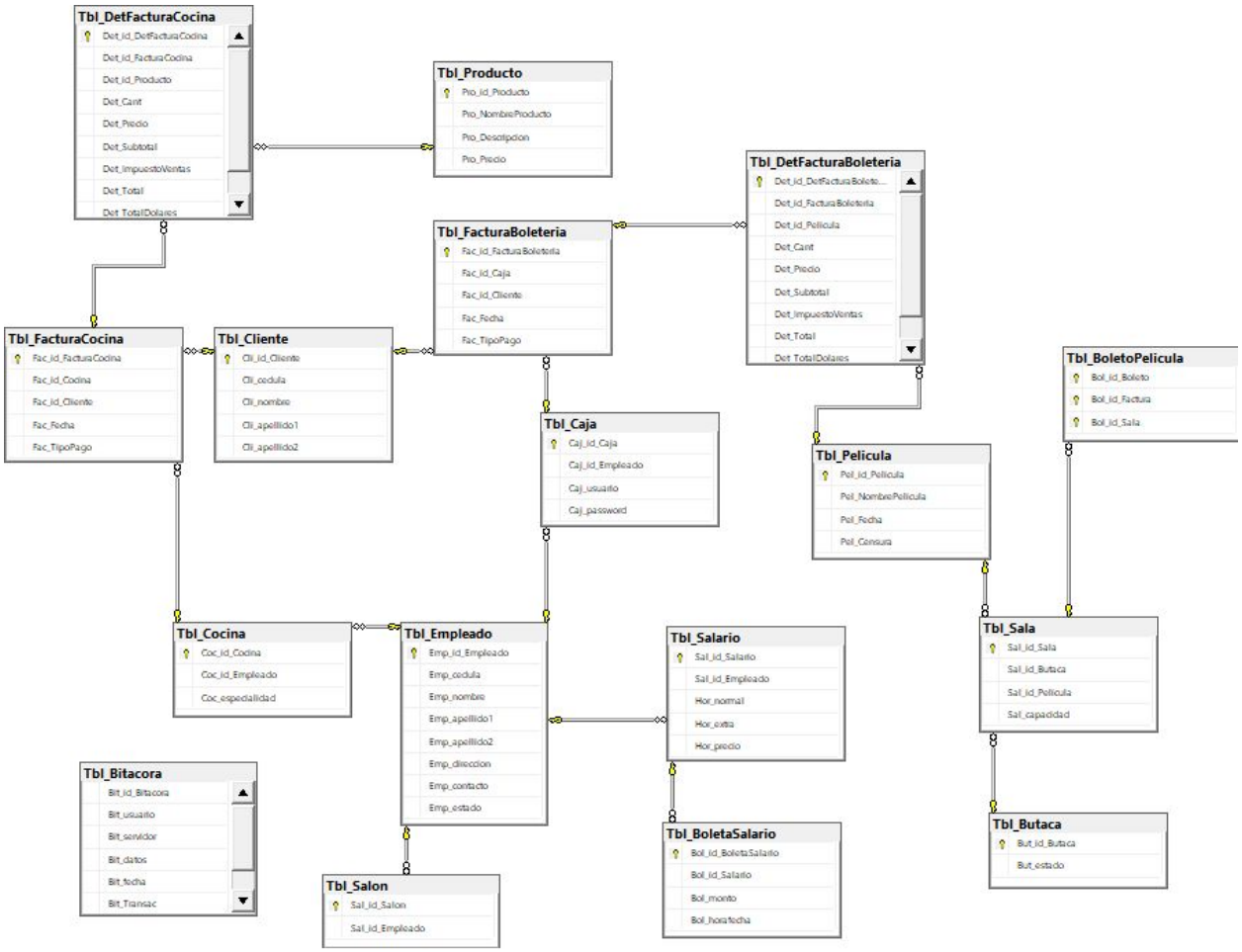
El detalle de la factura está ligada a una Boletería con su ID, el identificador de la caja , el identificador del cliente, la fecha y el tipo de campo, la boletería está ligada a una caja.

El cliente adquiere una factura que es generada por la boletería, la cédula será el identificador del cliente, el nombre es otro atributo de la entidad cliente, apellido uno y apellido dos. Además el cliente del cine puede generar facturas de lo consumido en la cocina de la sala de cines, las facturas cuenta su id, el id de la cocina en la sala de cine, el id del cliente, la fecha y el tipo de pago.

La entidad cocina genera facturas que a su vez cuenta con detalles de la facturación como su id del detalle de la factura, el id de la factura, el id del producto, la cantidad, el precio el subtotal, el impuesto de venta a lo consumido, y el total a pagar. Todas los detalles de las facturas contiene la especificaciones de los atributos de los producto como el id del producto ,el nombre del producto, una descripción y un precio.

Además todos los movimientos de divisas quedaran registrado en una entidad bitácora, que tendrá su propia ID, un usuario , estará vinculado a un servidor, almacenará datos, con su fecha y transacción.

Modelo Entidad - Relación



Esquema de seguridad

Control de acceso

Administrador

1. AdminPrincipal
    a. Tiene control de acceso a toda la base de datos
2. AdminSecundario
    a. Tiene control de acceso a toda la base de datos

Jefe

1. Tienes permiso de insertar, actualizar y borrar empleados; así como para ingresar a los empleados en tres categoríasg

Tablas	Procedimientos	
Tbl_Empleado Tbl_Caja Tbl_Cocina Tbl_Salon Tbl_Salario	Sp_Ins_Empleado Sp_Del_Empleado Sp_Upd_Empleado Sp_Ins_Salario	

Caja

1. EmpleadoCaja
    a. Tiene permiso para insertar, actualizar y borrar facturas de boletería

Tablas	Procedimientos	
Tbl_FacturaBoleteria Tbl_DetFacturaBoleteria	Sp_Ins_FacturaBoleteria Sp_Ins_DetFacBoleteria	

Cocina

1. EmpleadoCocina
    a. Tiene permiso para insertar, actualizar y borrar facturas de cocina

Tablas	Procedimientos	
Tbl_FacturaCocina Tbl_DetFacturaCocina	Sp_Ins_FacturaCocina Sp_Ins_DetFacCocina	

Auditorías

1. AuditorCines
    a. Tiene permiso para realizar las auditorías dentro de la empresa

Tablas	Procedimientos	
Tbl_Bitacora	Sp_Ins_Bitacora	

Plan de Respaldo

- El respaldo de información se efectuará en el ordenador del administrador secundario [AdminSecundario] integrante del grupo de DBA .
- El respaldo de la base de datos se realizará diariamente antes de las 11:00 PM.
- Los Respaldos se realizan por día, semana y mensual. En el caso de que no se puedan hacer los respaldos por algún problema con el Servidor (Virus o falla en unidad de almacenamiento DAT´s) se procede a realizarlos al día siguiente.
- Se hará el respaldo de lunes a jueves con etiqueta de la fecha de respaldo. El respaldo correspondiente al viernes se realizará conteniendo la información de la semana.

  Ejemplo:

  1ª. - Semana ---------- Viernes 1

  2ª. - Semana ---------- Viernes 2

  3ª. - Semana ---------- Viernes 3

  4ª. - Semana ----------Mes 1

- Debido a que es un sistema de tiempo compartido la solución que se ajusta a las necesidades es realizar respaldos dinámicos o respaldos en caliente porque no requieren tiempo de inactividad.

Diccionario de la base de datos

[Tbl_Empleado]

La Tabla Empleado , es para guardar la información de todos los empleados que pertenece al cine CCM Cinema, tiene siete atributos y son los siguientes:

- Emp_id_Empleado int NOT NULL,
- Emp_cedula int NOT NULL,
- Emp_nombre varchar(45) NOT NULL,
- Emp_apellido1 varchar(45) NOT NULL,
- Emp_apellido2 varchar(45) NOT NULL,
- Emp_direccion varchar(100) NOT NULL,
- Emp_contacto varchar(100) NOT NULL,
- Emp_estado char NOT NULL DEFAULT 1

Justificación de los atributos

La columna “Emp_id_Empleado” es la Primary Key, este es el dato que permite identificar a cada uno de los trabajadores y así no empleados repetidos y dos empleados con el mismo identificador. Este atributo es de tipo int y nunca puede estar este campo vacío, es un dato obligatorio.

“Emp_cedula” es para conocer cómo es identificado el trabajador en el sistema de Costa Rica, y así poder realizar procesos de asegurado, o a la hora de depositar su salario, entre otros procesos que sean necesarios. Es de tipo int y tiene como característica un “NOT NULL”

“Emp_nombre” este campo guarda solo el primer nombre del trabajador, para así tener información de como llamarlo, es más sencillo conocerlo por su nombre que por un número, además en documentos como factura tendrá esta información. Es de tipo varchar de tamaño de 45 y no puede estar vacío.

“Emp_apellido1” Este atributo forma parte del nombre completo del trabajador, es para guardar el primer apellido. Es de tipo varchar de 45 y es un not null.

“Emp_apellido2” Este atributo forma parte del nombre completo del trabajador, es para guardar el segundo apellido. Es de tipo varchar 45 y es un not null.

NOTA: No hay un campo para el segundo nombre del trabajador, se cree q es innecesario mantener crear un espacio en la Base de Datos, porque una persona es bien identificada con su cédula, nombre, apellido1 y apellido2.

“Emp_direccion” Este dato es para saber en dónde se podría localizar a esta persona que nos da sus servicios, en caso de alguna situación. Es de tipo varchar 100 y es un not null.

“Emp_contacto” Para almacenar un número telefónico por alguna situación.Es de tipo varchar 100 y no es null.

“Emp_estado” Es de tipo char NOT NULL DEFAULT 1. Es para no borrar el empleado de la Base de Datos, si está en “cero” está inactivo y está en “uno” está activo. Cuando el empleado ya no trabaja en la empresa se actualiza la columna estado a un cero para no borrarlo de todo del registro, porque en un futuro podrían pedir un listado histórico de los empleados de la empresa.

[Tbl_Salario]

Tiene una primary key compuesta. El espacio Salario se hizo con el fin de tener un control del salario que le corresponde a cada uno de los trabajadores del cine CCM Cinema, sus atributos son los siguientes:

- Sal_id_Salario int IDENTITY(1,1) NOT NULL,
- Sal_id_Empleado int NOT NULL,
- Hor_normal int NOT NULL,
- Hor_extra int NOT NULL,
- Hor_precio decimal(28,2) NOT NULL

Justificación de los atributos

“Sal_id_Salario” Este atributo es para tener un número que identifique a todos los salario,. Es de tipo int IDENTITY(1,1) NOT NULL, este atributo son números y ese número va a incrementar de forma automática, a partir del número uno y va ir creciendo de uno en uno, este proceso se da por el (1,1) que tiene en sus características.

“Sal_id_Empleado” este otro atributo forma parte de los Primary Key de la tabla Salario, por lo que es la forma de conocer el salario que tiene cierto trabajador porque con este primary key se relaciona con otra tabla llamada tbl_Empleado.

“Hor_normal” Para poder almacenar las horas normales que trabaja por quincena, para poder calcular su salario. Este es de tipo int y no puede estar vacío.

“Hor_extra” Algunos trabajadores realizan horas extras, las cuales tienen que ser remuneradas. Es de tipo int y no puede estar vacío,

“Hor_precio” Este atributo guarda de precio que tiene la hora de trabajo de cierto empleado, porque esto depende de su puesto de trabajo. Es de tipo decimal y tiene 28

caracteres con 2 decimales y es un campo que de forma obligatoria tiene q tener información por lo que es un “not null”

[Tbl_BoletaSalario]

Esta boleta es generada con un trigger, y cada trabajador tiene la suya por lo que se necesita tener conexión con la tabla Salario, esta tiene información del salario y del Empleado. Los atributos pertenecientes a la correspondiente tabla son:

- Bol_id_BoletaSalario int IDENTITY(1,1) NOT NULL,
- Bol_id_Salario int NOT NULL,
- Bol_monto decimal(28,2),
- Bol_horafecha datetime NOT NULL DEFAULT GETDATE(),

Justificación de los atributos

“Bol_id_BoletaSalario” Es tipo int, IDENTITY por lo que aumentará de forma automáticamente y (1,1) esto significa que iniciará su identificación a partir del número uno y aumentará de uno en uno y es un campo que tiene que tener información de forma obligatoria es decir es un “NOT NULL”

“Bol_id_Salario” este atributo es para que sea la foreign key con la tabla salario. Es de tipo int NOT NULL.

“Bol_monto” Este atributo es para tener un lugar en donde poder almacenar el total del salario de un trabajador. Es de tipo decimal(28,2).

“Bol_horafecha” Es de tipo datetime NOT NULL DEFAULT GETDATE(), para identificar el momento en que se generó este proceso, la fecha será por default, por lo que será la fecha que tiene el servidor.

[Tbl_Salon]

Es una tabla hija de la tabla empleado, es para tener un registro de las personas que trabajan en esa área, porque así se categoriza el salario de ese trabajador. Esta tabla es pensando a nivel de aplicación para que a la hora de registrar un trabajador y colocarle su àrea de labor, esa información se almacene en esta tabla. Esta tabla está compuesta por los siguientes atributos:

- Sal_id_Lobby int NOT NULL,
- Sal_id_Empleado int NOT NULL

Justificación de los atributos

“Sal_id_Lobby” Este atributo es de tipo int NOT NULL, él cuál no será autoincremental.
“Sal_id_Empleado” Este atributo es de tipo int NOT NULL para que funcione como foreign key con la tabla empleado.

[Tbl_Caja]

Es una tabla hija de la tabla empleado, es para identificar a los trabajadores que se encuentran como cajeros de boletería, esta se encuentra compuesta por los siguientes atributos.

- Caj_id_Caja int NOT NULL,
- Caj_id_Empleado int NOT NULL,
- Caj_usuario varchar(32) NOT NULL,
- Caj_password varchar(32) NOT NULL

Justificación de los atributos

“Caj_id_Caja” Es de tipo int NOT NULL y no es autoincremental.

“Caj_id_Empleado” Es de tipo int NOT NULL, es para que funcione como foreign key con la tabla empleados, para conocer el trabajador que está en caja.

“Caj_usuario” Es de tipo varchar(32) NOT NULL, para que el trabajador que está en caja tenga un usuario, porque para poder usar el programa tiene que tener un usuario.

“Caj_password” Es de tipo varchar(32) NOT NULL, se guarda la contraseña del trabajador que está en caja, para que pueda usar el programa que le ayuda a hacer su trabajo en boletería.

[Tbl_Cocina]

Esta tabla es para tener los trabajadores que se encuentran en el área de comida de la sala de cine ya sean en caja o sirviendo los pedidos, y tiene los siguientes atributos:

- Coc_id_Cocina int NOT NULL,
- Coc_id_Empleado int NOT NULL,
- Coc_especialidad varchar(45) NOT NULL

Justificación de los atributos

“Coc_id_Cocina” Esta es la Primary Key de la tabla cocina y es un entero y no puede estar vacía.

“Coc_id_Empleado” Este atributo tiene la función de foreign key y es de tipo int NOT NULL.

“Coc_especialidad” Es de tipo varchar(45) NOT NULL. aquí se identifica si el trabajador está en caja o sirviendo los pedidos.

[Tbl_FacturaCocina]

Esta tabla es para tener almacenado el encabezado de todas las facturas que pertenece a la caja del área de comida y tiene los siguientes atributos:

- Fac_id_FacturaCocina int NOT NULL,
- Fac_id_Cocina int NOT NULL,
- Fac_id_Cliente int NOT NULL,
- Fac_Fecha datetime NOT NULL DEFAULT getdate(),
- Fac_TipoPago varchar(45) NOT NULL

Justificación de los atributos

“Fac_id_FacturaCocina” Es de tipo int NOT NULL, funciona como primary key.
“Fac_id_Cocina” Es de tipo int NOT NULL y está hecha como foreign key, con la tabla Cocina para identificar de dónde ha sido generada la factura.
“Fac_id_Cliente” Es de tipo int NOT NULL, está hecha para que sea foreign key con la tabla cliente para que la factura tenga este registro.
“Fac_Fecha” Es de tipo datetime NOT NULL DEFAULT getdate(), para que la factura tenga la fecha y la hora en que fue generada y así tener los datos exactos del momento de la venta.
“Fac_TipoPago” Es de tipo varchar(45) NOT NULL, para controlar si el pago fue a contado o con tarjeta.

[Tbl_DetFacturaCocina]

Es la tabla que tiene el detalle de la factura generada en el área de comida, tiene los siguientes atributos:

- Det_id_DetFacturaCliente int NOT NULL,
- Det_id_FacturaCocina int NOT NULL,
- Det_id_Producto int NOT NULL,
- Det_Cant smallint NOT NULL,
- Det_Precio decimal(20,2) NOT NULL,
- Det_Subtotal decimal(20,2) NOT NULL,
- Det_ImpuestoVentas decimal(20,2) NOT NULL,
- Det_Total decimal(20,2) NOT NULL,
- Det_TotalDolares decimal(20,2) NOT NULL

Justificación de los atributos

“Det_id_DetFacturaCliente int NOT NULL, Det_id_FacturaCocina, Det_id_Producto” Estos tres atributos son primary key, porque para tener detalles de la factura se necesita información de la tabla factura cocina y de la tabla producto.
“Det_Cant “ es un atributo que guarda la cantidad de los productos que compran y no se necesita de tantos número, port eso su tipo es smallint.
“Det_Precio” Es de tipo decimal(20,2) NOT NULL, y es para guardar el precio de cada cosa que compren.
“Det_Subtotal” Es de tipo decimal(20,2) NOT NULL, para guardar el subtotal de la compra sin ningún tipo de impuesto, es el precio bruto.
“Det_ImpuestoVentas” Es de tipo decimal(20,2) NOT NULL, es para colocar el impuestos de ventas.
“Det_Total” Es de tipo decimal(20,2) NOT NULL, es para guardar el valor neto de la compra.
“Det_TotalDolares” Es de tipo decimal(20,2) NOT NULL y es para guardar el valor neto en dólares de la compra.

[Tbl_FacturaBoleteria]

Es para guardar la información de las facturas que se genera en el área de boletería, pero para el encabezado de la factura y tiene los siguientes atributos:

- Fac_id_FacturaBoleteria int NOT NULL,
- Fac_id_Caja int NOT NULL,
- Fac_id_Cliente int NOT NULL,
- Fac_Fecha datetime NOT NULL DEFAULT getdate(),
- Fac_TipoPago varchar(45) NOT NULL

Justificación de los atributos

“Fac_id_FacturaBoleteria, Fac_id_Caja, Fac_id_Cliente” Estos tres atributos son las Primary Keys de la respectiva tabla.

“Fac_Fecha” Es de tipo datetime NOT NULL DEFAULT getdate(), y es para tener la fecha y hora en que se generó la venta.

“Fac_TipoPago “ Para controlar si el pago fue a contado o con tarjeta.

[Tbl_DetFacturaBoleteria]

Es la tabla que tiene el detalle de la factura generada en el área de boletería, tiene los siguientes atributos:

- Det_id_DetFacturaBoleteria int NOT NULL,
- Det_id_FacturaCaja int NOT NULL,
- Det_id_Producto int NOT NULL,
- Det_Cant smallint NOT NULL,
- Det_Precio decimal(20,2) NOT NULL,
- Det_Subtotal decimal(20,2) NOT NULL,

- Det_ImpuestoVentas decimal(20,2) NOT NULL,
- Det_Total decimal(20,2) NOT NULL,
- Det_TotalDolares decimal(20,2) NOT NULL

Justificación de los atributos

“Det_id_DetFacturaBoleteria, Det_id_FacturaCaja int NOT NULL, Det_id_Producto” Son las Primary Keys de la respectiva tabla.

“Det_Cant “ es un atributo que guarda la cantidad de boletos que compran y no se necesita de tantos número, port eso su tipo es smallint.

“Det_Precio” Es de tipo decimal(20,2) NOT NULL, y es para guardar el precio de la película.

“Det_Subtotal” Es de tipo decimal(20,2) NOT NULL, para guardar el subtotal de la venta sin ningún tipo de impuesto, es el precio bruto.

“Det_ImpuestoVentas” Es de tipo decimal(20,2) NOT NULL, es para colocar el impuestos de ventas.

“Det_Total” Es de tipo decimal(20,2) NOT NULL, es para guardar el valor neto de la venta.

“Det_TotalDolares” Es de tipo decimal(20,2) NOT NULL y es para guardar el valor neto en dólares de la compra.

[Tbl_Producto]

Es la tabla que guarda toda información de los productos que maneja la empresa y tiene los siguientes atributos:

- Pro_id_Producto varchar (45) NOT NULL,
- Pro_NombreProducto varchar(45) NOT NULL,
- Pro_Descripcion varchar(90) NULL,
- Pro_Precio smallint NOT NUll

Justificación de los atributos

“Pro_id_Producto” Es de tipo varchar (45) NOT NULL, es la Primary Key de la respectiva tabla.

“Pro_NombreProducto” Es de tipo varchar(45) NOT NULL, es para guardar el nombre del producto comestible o el nombre del combo

“Pro_Descripcion” Es de tipo varchar(90) NULL, lo que lleva el combo y si no es un combo, no se coloca ningún tipo de descripción

“Pro_Precio” Es de tipo smallint NOT NUll, para guardar el precio del producto o el combo.

[Tbl_Cliente]

Para guardar la información de los clientes de la empresa, tiene los siguientes atributos:

- Cli_id_Cliente int NOT NULL,
- Cli_cedula int NOT NULL,
- Cli_nombre varchar(45) NOT NULL,
- Cli_apellido1 varchar(45) NOT NULL,
- Cli_apellido2 varchar(45) NOT NULL,

Justificación de los atributos

“Cli_id_Cliente” Es de tipo int NOT NULL. Es la Primary key de la respectiva tabla.

“Cli_cedula” Es de tipo int NOT NULL. Para guardar la cédula del cliente, para poder identificarlo.

“Cli_nombre, Cli_apellido1, Cli_apellido2” Es para poder obtener el nombre completo del cliente, sin tomar en cuenta el segundo nombre.

[Tbl_BoletoPelicula]

Esta tabla es para guardar la información que llega a generar la factura de boletería, porque si hay una factura en ea área esto provoca un boleto, en el cual se encuentra la siguiente información:

- Bol_id_Boleto int NOT NULL,
- Bol_id_Factura int NOT NULL,
- Bol_id_Sala int NOT NULL,
- **Bol_id_Butaca varchar (3) NOT NULL**

Justificación de los atributos

“Bol_id_Boleto, Bol_id_Factura, Bol_id_Sala” Estos tres atributos son Primarys Keys de la respectiva tabla.

[Tbl_Sala]

Es para tener la información de cada sala, y esa información es la siguiente:

- Sal_id_Sala int NOT NULL,
- Sal_id_Butaca int NOT NULL,
- Sal_id_Pelicula int NOT NULL,
- Sal_capacidad smallint NOT NULL

Justificación de los atributos

“Sal_id_Sala, Sal_id_Butaca, Sal_id_Pelicula” Los tres atributos son la Primary Key “Sal_capacidad” Es de tipo smallint NOT NULL, para tener la capacidad que tiene cada sala.

[Tbl_Pelicula]

En esta tabla se encuenta información de la película que se encuentra en la sucursal y son los siguientes datos:

- Pel_id_Pelicula int NOT NULL,

- Pel_NombrePelicula varchar(45) NOT NULL,
- Pel_Fecha datetime NOT NULL DEFAULT getdate(),
- Pel_Censura varchar(45) NOT NULL,

Justificación de los atributos

“Pel_id_Pelicula” Es de tipo int NOT NULL y es la Primary Key de la respectiva tabla.
“Pel_NombrePelicula” Es de tipo varchar(45) NOT NULL, ahí se encuentra el nombre de la película
“Pel_Fecha” Es de tipo datetime NOT NULL DEFAULT getdate(), es para guardar la fecha y la hora en entró en el registro.
“Pel_Censura” Es de tipo varchar(45) NOT NULL, para registrar si es para todo público, mayores de 18 años, mayores de 12 años o la censura que tiene.

[Tbl_Butaca]

En esta tabla se encuentra la información de las butacas de la sala de cine, y son los siguientes atributos:

- But_id_Butaca varchar(3) NOT NULL,
- But_estado char NOT NULL

Justificación de los atributos

“But_id_Butaca” Es de tipo varchar(3) NOT NULL. Este atributo es la Primary Key de la respectiva tabla.

“But_estado” Es de tipo char NOT NULL. Este atributo es para saber si está ocupado o desocupado. Se identifica de la siguiente forma : ‘O’ = OCUPADO / ‘D’ = DESOCUPADO

[Tbl_Bitacora]

Esta tabla es para guardar los datos de los disparadores (Triggers). Se encuentra construida con los siguientes atributos:

- Bit_id_Bitacora int NOT NULL,
- Bit_usuario varchar(45) NOT NULL,
- Bit_servidor varchar(45) NOT NULL,
- Bit_datos varchar(max) NOT NULL,
- Bit_fecha datetime NOT NULL DEFAULT GETDATE(),
- Bit_Transac char

Justificación de los atributos

“Bit_id_Bitacora” Es de tipo int NOT NULL. Es el atributo que funciona como Primary Key.
“Bit_usuario” Es de tipo varchar(45) NOT NULL. Es para registrar el usuario que ejecutó una acción en cierta tabla.
“Bit_servidor” Es de tipo varchar(45) NOT NULL. Es para guardar el nombre del servidor en donde se ejecutó el trigger.
“Bit_datos” Es de tipo varchar(max) NOT NULL. Es para guardar todos los datos que se ingresaron, actualizaron o se eliminaron en cierta tabla en donde se ejecuta el trigger.
“Bit_fecha” Es de tipo datetime NOT NULL DEFAULT GETDATE(),. Es para registrar la fecha y la hora en que se ejecuta el trigger, este dato se da de forma automática y coloca la fecha y hora que tiene el servidor en donde se ejecutó la acción.
“Bit_Transac” Es de tipo char. Es para identificar cuál fue la acción el trigger, por ejemplo:

‘I’ = Insert

‘U’ = Update

‘D’ = Delete

Scripts del Modelo Entidad Relación

TALES.SQL

```
USE master
GO

CREATE DATABASE BD_CineRocko
ON
(
	NAME = db_Occidente_dat
	,FILENAME = 'C:\CinemaRocko\db_Occidente_MDF.mdf'
	,SIZE = 100MB
	,MAXSIZE = 1000MB
	,FILEGROWTH = 15%
)
LOG ON
(
	NAME = db_Occidente_log
	,FILENAME = 'C:\CinemaRocko\db_Occidente_LDF.ldf'
	,SIZE = 15MB
	,MAXSIZE = 50MB
	,FILEGROWTH = 5MB
);
GO
```

```
USE BD_CineRocko
GO

--TABLA PARA LOS DATOS DEL EMPLEADO
-------------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tbl_Empleado')
DROP TABLE [Tbl_Empleado]
GO
CREATE TABLE [Tbl_Empleado]
(
        Emp_id_Empleado int NOT NULL,
        Emp_cedula int NOT NULL,
        Emp_nombre varchar(45) NOT NULL,
        Emp_apellido1 varchar(45) NOT NULL,
        Emp_apellido2 varchar(45) NOT NULL,
        Emp_direccion varchar(100) NOT NULL,
        Emp_contacto varchar(100) NOT NULL
);

-- SE ALTERA LA TABLA AÑADIENDO LA COLUMNA ESTADO
ALTER  TABLE [dbo].[Tbl_Empleado]
ADD Emp_estado char DEFAULT 1;

-- TABLA PARA LOS DATOS DE SALARIO
-------------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Tbl_Salario')
DROP TABLE [Tbl_Salario]
GO
```

```
CREATE TABLE [Tbl_Salario]
(
	Sal_id_Salario int IDENTITY(1,1) NOT NULL,
	Sal_id_Empleado int NOT NULL,
	Hor_normal int NOT NULL,
	Hor_extra int NOT NULL,
	Hor_precio decimal(28,2) NOT NULL,
);

-- TABLA PARA LOS DATOS DE LA BOLETA DE PAGO
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tbl_BoletaSalario')
DROP TABLE [Tbl_BoletaSalario]
GO
CREATE TABLE [Tbl_BoletaSalario]
(
	Bol_id_BoletaSalario int IDENTITY(1,1) NOT NULL,
	Bol_id_Salario int NOT NULL,
	Bol_monto decimal(28,2),
	Bol_horafecha datetime NOT NULL DEFAULT GETDATE(),
);

-- TABLA PARA LA CATEGORIA DE EMPLEADOS DE SALON
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Tbl_Salon')
DROP TABLE [Tbl_Salon]
GO
CREATE TABLE[Tbl_Salon]
```

```
(
	Sal_id_Salon int NOT NULL,
	Sal_id_Empleado int NOT NULL
);

-- TABLA PARA LA SUBCATEGORIA EMPLEADOS DE CAJA
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Tbl_Caja')
DROP TABLE [Tbl_Caja]
GO
CREATE TABLE[Tbl_Caja]
(
	Caj_id_Caja int NOT NULL,
	Caj_id_Empleado int NOT NULL,
	Caj_usuario varchar(32) NOT NULL,
	Caj_password varchar(32) NOT NULL
);

-- TABLA PARA LA SUBCATEGORIA DE COCINA
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Tbl_Cocina')
DROP TABLE [Tbl_Cocina]
GO
CREATE TABLE[Tbl_Cocina]
(
	Coc_id_Cocina int NOT NULL,
	Coc_id_Empleado int NOT NULL,
	Coc_especialidad varchar(45) NOT NULL
);
```

```
-- TABLA PARA LA FACTURA DE ALIMENTOS
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tbl_FacturaCocina')
DROP TABLE [Tbl_FacturaCocina]
GO
CREATE TABLE [Tbl_FacturaCocina]
(
        Fac_id_FacturaCocina int NOT NULL,
        Fac_id_Cocina int NOT NULL,
        Fac_id_Cliente int NOT NULL,
        Fac_Fecha datetime NOT NULL DEFAULT getdate(),
        Fac_TipoPago varchar(45) NOT NULL
);

-- TABLA PARA EL DETALLE DE FACTURA DE ALIMENTOS
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM SYS.OBJECTS WHERE NAME =
'Tbl_DetFacturaCocina')
DROP TABLE [Tbl_DetFacturaCocina]
GO
CREATE TABLE [Tbl_DetFacturaCocina]
(
        Det_id_DetFacturaCocina int NOT NULL,
        Det_id_FacturaCocina int NOT NULL,
        Det_id_Producto int NOT NULL,
        Det_Cant smallint NOT NULL,
        Det_Precio decimal(20,2) NOT NULL,
```

```
	Det_Subtotal decimal(20,2) NOT NULL,
	Det_ImpuestoVentas decimal(20,2) NOT NULL,
	Det_Total decimal(20,2) NOT NULL,
	Det_TotalDolares decimal(20,2) NOT NULL
);

-- TABLA PARA LA FACTURA DE BOLETERIA
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tbl_FacturaBoleteria')
DROP TABLE [Tbl_FacturaBoleteria]
GO
CREATE TABLE [Tbl_FacturaBoleteria]
(
	Fac_id_FacturaBoleteria int NOT NULL,
	Fac_id_Caja int NOT NULL,
	Fac_id_Cliente int NOT NULL,
	Fac_Fecha datetime NOT NULL DEFAULT getdate(),
	Fac_TipoPago varchar(45) NOT NULL
);

-- TABLA PARA EL DETALLE DE FACTURA DE BOLETERIA
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM SYS.OBJECTS WHERE NAME =
'Tbl_DetFacturaBoleteria')
DROP TABLE [Tbl_DetFacturaBoleteria]
GO
CREATE TABLE [Tbl_DetFacturaBoleteria]
(
```

```
	Det_id_DetFacturaBoleteria int NOT NULL,
	Det_id_FacturaBoleteria int NOT NULL,
	Det_id_Pelicula int NOT NULL,
	Det_Cant smallint NOT NULL,
	Det_Precio decimal(20,2) NOT NULL,
	Det_Subtotal decimal(20,2) NOT NULL,
	Det_ImpuestoVentas decimal(20,2) NOT NULL,
	Det_Total decimal(20,2) NOT NULL,
	Det_TotalDolares decimal(20,2) NOT NULL
);

-- TABLA PARA ALMACENAR LA INFORMACION DE PRODUCTOS
-----------------------------------------------------------------------------------------------
IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE
NAME='Tbl_Producto')
DROP TABLE [Tbl_Producto]
GO
CREATE TABLE [Tbl_Producto]
(
	Pro_id_Producto int NOT NULL,
	Pro_NombreProducto varchar(45) NOT NULL,
	Pro_Descripcion  varchar(90) NOT NULL,
	Pro_Precio smallint NOT NUll
);

-- TABLA PARA ALMACENAR CLIENTES
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Tbl_Cliente')
DROP TABLE [Tbl_Cliente]
```

```
GO
CREATE TABLE [Tbl_Cliente]
(
	Cli_id_Cliente int NOT NULL,
	Cli_cedula int NOT NULL,
	Cli_nombre varchar(45) NOT NULL,
	Cli_apellido1 varchar(45) NOT NULL,
	Cli_apellido2 varchar(45) NOT NULL,
);

-- TABLA PARA EL BOLETO DE LA PELICULA
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tbl_BoletoPelicula')
DROP TABLE [Tbl_BoletoPelicula]
GO
CREATE TABLE [Tbl_BoletoPelicula]
(
	Bol_id_Boleto int NOT NULL,
	Bol_id_Factura int NOT NULL,
	Bol_id_Sala int NOT NULL
);

-- TABLA PARA LA SALA
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Tbl_Sala')
DROP TABLE [Tbl_Sala]
GO
```

```
CREATE TABLE [Tbl_Sala]
(
	Sal_id_Sala int NOT NULL,
	Sal_id_Butaca int NOT NULL,
	Sal_id_Pelicula int NOT NULL,
	Sal_capacidad smallint NOT NULL
);

-- TABLA PARA LA PELICULA
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tbl_Pelicula')
DROP TABLE [Tbl_Pelicula]
GO
CREATE TABLE [Tbl_Pelicula]
(
	Pel_id_Pelicula int NOT NULL,
	Pel_NombrePelicula varchar(45) NOT NULL,
	Pel_Fecha datetime NOT NULL DEFAULT getdate(),
	Pel_Censura varchar(45) NOT NULL,
);

-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Tbl_Butaca')
DROP TABLE [Tbl_Butaca]
GO

-- TABLA PARA LA BITACORA
CREATE TABLE [Tbl_Butaca]
```

```
(
	But_id_Butaca int NOT NULL,
	But_estado char NOT NULL
);

-- TABLA PARA LA BITACORA
-----------------------------------------------------------------------------------------------
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tbl_Bitacora')
DROP TABLE [Tbl_Bitacora]
GO
CREATE TABLE [Tbl_Bitacora]
(
	Bit_id_Bitacora int IDENTITY(1,1) NOT NULL,
	Bit_usuario varchar(45) NOT NULL,
	Bit_servidor varchar(45) NOT NULL,
	Bit_datos varchar(max) NOT NULL,
	Bit_fecha datetime NOT NULL DEFAULT GETDATE(),
	Bit_Transac char
);
```

CONSTRAINT.SQL

```
USE BD_CineRocko
GO

--Llave primaria [Tbl_Empleado]
ALTER TABLE [dbo].[Tbl_Empleado]
ADD CONSTRAINT Pk_Empleados
PRIMARY KEY([Emp_id_Empleado]);

--Llave primaria [Tbl_Cocina]
ALTER TABLE [dbo].[Tbl_Cocina]
ADD CONSTRAINT Pk_Cocina
PRIMARY KEY([Coc_id_Cocina]);

--Llave primaria [Tbl_Caja]
ALTER TABLE [dbo].[Tbl_Caja]
ADD CONSTRAINT Pk_Caja
PRIMARY KEY([Caj_id_Caja]);

--Llave primaria [Tbl_Salon]
ALTER TABLE [dbo].[Tbl_Salon]
ADD CONSTRAINT Pk_Salon
PRIMARY KEY([Sal_id_Salon]);

--Llave primaria [Tbl_Salario]
ALTER TABLE [dbo].[Tbl_Salario]
ADD CONSTRAINT Pk_Salario
```

```
PRIMARY KEY([Sal_id_Salario]);

--Llave primaria [Tbl_BoletaSalario]
ALTER TABLE [dbo].[Tbl_BoletaSalario]
ADD CONSTRAINT Pk_BoletaSalario
PRIMARY KEY([Bol_id_BoletaSalario]);

--Llave primaria [Tbl_Cliente]
ALTER TABLE [dbo].[Tbl_Cliente]
ADD CONSTRAINT Pk_Cliente
PRIMARY KEY([Cli_id_Cliente]);

--Llave primaria [Tbl_Producto]
ALTER TABLE [dbo].[Tbl_Producto]
ADD CONSTRAINT Pk_Producto
PRIMARY KEY([Pro_id_Producto]);

--Llave primaria [Tbl_Sala]
ALTER TABLE [dbo].[Tbl_Sala]
ADD CONSTRAINT Pk_Sala
PRIMARY KEY([Sal_id_Sala]);

--Llave primaria [Tbl_Pelicula]
ALTER TABLE [dbo].[Tbl_Pelicula]
ADD CONSTRAINT Pk_Pelicula
PRIMARY KEY([Pel_id_Pelicula]);

--Llave primaria [Tbl_Butaca]
ALTER TABLE [dbo].[Tbl_Butaca]
```

```
ADD CONSTRAINT Pk_Butaca
PRIMARY KEY([But_id_Butaca]);

--Llave primaria [Tbl_BoletoPelicula]
ALTER TABLE [dbo].[Tbl_BoletoPelicula]
ADD CONSTRAINT Pk_BoletaPelicula
PRIMARY KEY([Bol_id_Boleto],[Bol_id_Factura],[Bol_id_Sala]);

--Llave primaria [Tbl_FacturaCocina]
ALTER TABLE [dbo].[Tbl_FacturaCocina]
ADD CONSTRAINT Pk_FacturaCocina
PRIMARY KEY([Fac_id_FacturaCocina]);

--Llave primaria [Tbl_DetFacturaCocina]
ALTER TABLE [dbo].[Tbl_DetFacturaCocina]
ADD CONSTRAINT Pk_DetFacturaCocina
PRIMARY KEY([Det_id_DetFacturaCocina]);

--Llave primaria [Tbl_FacturaBoleteria]
ALTER TABLE [dbo].[Tbl_FacturaBoleteria]
ADD CONSTRAINT Pk_FacturaBoleteria
PRIMARY KEY([Fac_id_FacturaBoleteria]);

--Llave primaria [Tbl_DetFacturaBoleteria]
ALTER TABLE [dbo].[Tbl_DetFacturaBoleteria]
ADD CONSTRAINT Pk_DetFacturaBoleteria
PRIMARY KEY([Det_id_DetFacturaBoleteria]);

ALTER TABLE [dbo].[Tbl_Bitacora]
```

```
ADD CONSTRAINT Pk_Bitacora
PRIMARY KEY([Bit_id_Bitacora]);

--Foreign key
ALTER TABLE [dbo].[Tbl_Cocina]
ADD CONSTRAINT Fk_Coc_Emp
FOREIGN KEY ([Coc_id_Empleado])
REFERENCES [dbo].[Tbl_Empleado]([Emp_id_Empleado]);

ALTER TABLE [dbo].[Tbl_Caja]
ADD CONSTRAINT Fk_Caj_Emp
FOREIGN KEY ([Caj_id_Empleado])
REFERENCES [dbo].[Tbl_Empleado]([Emp_id_Empleado]);

ALTER TABLE [dbo].[Tbl_Salon]
ADD CONSTRAINT Fk_Sal_Emp
FOREIGN KEY (Sal_id_Empleado)
REFERENCES [dbo].[Tbl_Empleado]([Emp_id_Empleado]);

ALTER TABLE [dbo].[Tbl_Salario]
ADD CONSTRAINT Fk_Salario_Emp
FOREIGN KEY ([Sal_id_Empleado])
REFERENCES [dbo].[Tbl_Empleado]([Emp_id_Empleado]);

ALTER TABLE [dbo].[Tbl_BoletaSalario]
ADD CONSTRAINT Fk_BolSalario_Sal
FOREIGN KEY ([Bol_id_Salario])
REFERENCES [dbo].[Tbl_Salario]([Sal_id_Salario]);
```

```
ALTER TABLE [dbo].[Tbl_Sala]
ADD CONSTRAINT Fk_Sala_Butaca
FOREIGN KEY([Sal_id_Butaca])
REFERENCES [dbo].[Tbl_Butaca]([But_id_Butaca]);

ALTER TABLE [dbo].[Tbl_Sala]
ADD CONSTRAINT Fk_Sala_Pelicula
FOREIGN KEY([Sal_id_Pelicula])
REFERENCES [dbo].[Tbl_Pelicula]([Pel_id_Pelicula]);

ALTER TABLE [dbo].[Tbl_BoletoPelicula]
ADD CONSTRAINT Fk_BolPel_Sala
FOREIGN KEY([Bol_id_Sala])
REFERENCES [dbo].[Tbl_Sala]([Sal_id_Sala]);

--Llaves foráneas de Tbl_FacturaBoleteria

ALTER TABLE [dbo].[Tbl_FacturaBoleteria]
ADD CONSTRAINT Fk_DetFac_Caja
FOREIGN KEY([Fac_id_Caja])
REFERENCES [dbo].[Tbl_Caja]([Caj_id_Caja]);

ALTER TABLE [dbo].[Tbl_FacturaBoleteria]
ADD CONSTRAINT Fk_DetFac_Cliente
FOREIGN KEY([Fac_id_Cliente])
REFERENCES [dbo].[Tbl_Cliente]([Cli_id_Cliente]);

--LLave foránea de Tbl_DetFacturaBoleteria
```

```
ALTER TABLE [dbo].[Tbl_DetFacturaBoleteria]
ADD CONSTRAINT Fk_DetFac_FactBoleteria
FOREIGN KEY([Det_id_FacturaBoleteria])
REFERENCES [dbo].[Tbl_FacturaBoleteria]([Fac_id_FacturaBoleteria]);

ALTER TABLE [dbo].[Tbl_DetFacturaBoleteria]
ADD CONSTRAINT Fk_DetFac_Pelicula
FOREIGN KEY([Det_id_Pelicula])
REFERENCES [dbo].[Tbl_Pelicula]([Pel_id_Pelicula]);

--Llaves foráneas de Tbl_FacturaCocina

ALTER TABLE [dbo].[Tbl_FacturaCocina]
ADD CONSTRAINT Fk_DetFac_Cocina
FOREIGN KEY([Fac_id_Cocina])
REFERENCES [dbo].[Tbl_Cocina]([Coc_id_Cocina]);

ALTER TABLE [dbo].[Tbl_FacturaCocina]
ADD CONSTRAINT Fk_DetFacCoc_Cliente
FOREIGN KEY([Fac_id_Cliente])
REFERENCES [dbo].[Tbl_Cliente]([Cli_id_Cliente]);

--LLave foránea de Tbl_DetFacturaCocina

ALTER TABLE [dbo].[Tbl_DetFacturaCocina]
ADD CONSTRAINT Fk_DetFac_FactCocina
FOREIGN KEY([Det_id_FacturaCocina])
REFERENCES [dbo].[Tbl_FacturaCocina]([Fac_id_FacturaCocina]);
```

```
ALTER TABLE [dbo].[Tbl_DetFacturaCocina]
ADD CONSTRAINT Fk_DetFac_Producto
FOREIGN KEY([Det_id_Producto])
REFERENCES [dbo].[Tbl_Producto]([Pro_id_Producto]);
```

DATA.SQL

```
USE BD_CineRocko
GO

-- PROCEDIMIENTOS ALMACENADOS PARA Tbl_Empleado

--INSERT

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Ins_Empleado')
DROP PROCEDURE [Sp_Ins_Empleado]
GO
CREATE PROCEDURE [Sp_Ins_Empleado]
(
	@idEmpleado int,
	@cedula int,
	@nombre varchar(45),
	@apellido1 varchar(45),
	@apellido2 varchar(45),
	@direccion varchar(100),
	@contacto varchar(100)
)
AS
```

```
INSERT INTO [dbo].[Tbl_Empleado]
(
	Emp_id_Empleado,
	Emp_cedula,
	Emp_nombre,
	Emp_apellido1,
	Emp_apellido2,
	Emp_direccion,
	Emp_contacto
)
VALUES
(

@idEmpleado,@cedula,@nombre,@apellido1,@apellido2,@direccion,@contacto
);
GO

-- UPDATE

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Sp_Upd_Empleado')
DROP PROCEDURE [Sp_Upd_Empleado]
GO
CREATE PROCEDURE [Sp_Upd_Empleado]
(
	@idEmpleado int,
	@cedula int,
	@nombre varchar(45),
	@apellido1 varchar(45),
```

```
	@apellido2 varchar(45),
	@direccion varchar(100),
	@contacto varchar(100)
)
AS
UPDATE [dbo].[Tbl_Empleado]
SET
	Emp_cedula = @cedula,
	Emp_nombre = @nombre,
	Emp_apellido1 = @apellido1,
	Emp_apellido2 = @apellido1,
	Emp_direccion = @direccion,
	Emp_contacto = @contacto
WHERE Emp_id_Empleado = @idEmpleado;
GO

-- DELETE

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Sp_Del_Empleado')
DROP PROCEDURE [Sp_Del_Empleado]
GO
CREATE PROCEDURE [Sp_Del_Empleado]
(
	@Emp_id_Empleado int
)
AS
UPDATE [dbo].[Tbl_Empleado]
SET [Emp_estado] = 0
```

```
WHERE Emp_id_Empleado = @Emp_id_Empleado;
GO


-- PROCEDIMIENTO ALMACENADO PARA Tbl_Salon

-- INSERT

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Ins_Salon')
DROP PROCEDURE [Sp_Ins_Salon]
GO
CREATE PROCEDURE [Sp_Ins_Salon](
        @Sal_id_Salon int,
        @Sal_id_Empleado int
)
AS
INSERT INTO Tbl_Salon(Sal_id_Salon, Sal_id_Empleado)
VALUES (@Sal_id_Salon, @Sal_id_Empleado);
GO

-- DELETE

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Del_Salon')
DROP PROCEDURE [Sp_Del_Salon]
GO
CREATE PROCEDURE [Sp_Del_Salon](
                @Sal_id_Salon int
```

```
)
AS
DELETE FROM [Tbl_Salon]
WHERE Sal_id_Salon = @Sal_id_Salon;
GO

-- FUNCION PARA ENCRIPTAR DATOS
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Fn_Encriptar')
DROP FUNCTION [Fn_Encriptar]
GO
CREATE FUNCTION [Fn_Encriptar]
(
	@parametro VARCHAR(max) -- Parámetro de entrada
)
RETURNS VARCHAR(max)
AS
BEGIN
DECLARE @secreto VARCHAR(max); -- Variable
	SET @secreto = ENCRYPTBYPASSPHRASE('4856',@parametro); -- Asigna a Varible Parámetro de entrada
	RETURN @secreto; -- Regresa el valor encriptado
END
GO


-- PROCEDIMIENTOS ALMACENADOS PARA Tbl_Caja

-- INSERT
```

```
IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Ins_Caja')
DROP PROCEDURE [Sp_Ins_Caja]
GO
CREATE PROCEDURE [Sp_Ins_Caja](
	@Caj_id_Caja int,
	@Caj_id_Empleado int,
	@Caj_usuario varchar(32),
	@Caj_password varchar(32)
)
AS
INSERT INTO Tbl_Caja (Caj_id_Caja, Caj_id_Empleado, Caj_usuario, Caj_password)
VALUES (@Caj_id_Caja , @Caj_id_Empleado, @Caj_usuario, [dbo].[Fn_Encriptar](@Caj_password))
GO

-- UPDATE

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Upd_Caja')
DROP PROCEDURE [Sp_Upd_Caja]
GO
CREATE PROCEDURE [Sp_Upd_Caja](
	@Caj_id_Caja int,
	@Caj_usuario varchar(32),
	@Caj_password varchar(32)
)
AS
```

```
UPDATE [Tbl_Caja]
SET
	Caj_usuario= @Caj_usuario,
	Caj_password = @Caj_password
WHERE Caj_id_Caja = @Caj_id_Caja;
GO

-- DELETE

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Del_Caja')
DROP PROCEDURE [Sp_Del_Caja]
GO
CREATE PROCEDURE [Sp_Del_Caja](
	@Caj_id_Caja int)
AS
DELETE FROM [Tbl_Caja]
WHERE Caj_id_Caja = @Caj_id_Caja;
GO

-- PROCEDIMIENTOS ALMACENADOS PARA Tbl_Cocina

-- INSERT

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Ins_Cocina')
DROP PROCEDURE [Sp_Ins_Cocina]
GO
CREATE PROCEDURE [Sp_Ins_Cocina] (
```

```
	@Coc_id_Cocina int,
	@Coc_id_Empleado int,
	@Coc_especialidad varchar(45)
)
AS
INSERT INTO Tbl_Cocina (Coc_id_Cocina, Coc_id_Empleado, Coc_especialidad)
VALUES (@Coc_id_Cocina, @Coc_id_Empleado, @Coc_especialidad)
GO

-- UPDATE

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Sp_Upd_Cocina')
DROP PROCEDURE [Sp_Upd_Cocina]
GO
CREATE PROCEDURE [Sp_Upd_Cocina](
	@Coc_id_Cocina int,
	@Coc_especialidad varchar(45)
)
AS
UPDATE [Tbl_Cocina]
SET
	Coc_especialidad = @Coc_especialidad
WHERE Coc_id_Cocina = @Coc_id_Cocina;
GO

-- DELETE
```

```
IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Del_Cocina')
DROP PROCEDURE [Sp_Del_Cocina]
GO
CREATE PROCEDURE [Sp_Del_Cocina](
                @Coc_id_Cocina int)
AS
DELETE FROM Tbl_Cocina
WHERE Coc_id_Cocina = @Coc_id_Cocina;
GO

-- PROCEDIMIENTOS ALMACENADOS PARA Tbl_Salario

-- INSERT

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Ins_Salario')
DROP PROCEDURE [Sp_Ins_Salario]
GO
CREATE PROCEDURE [Sp_Ins_Salario]
(
        @id_Empleado int,
        @normal int,
        @extra int,
        @precio decimal(28,2)
)
AS
INSERT INTO [dbo].[Tbl_Salario]
(
```

```
	Sal_id_Empleado,Hor_normal,Hor_extra,Hor_precio
)
VALUES
(
	@id_Empleado,@normal,@extra,@precio
);
GO

-- PROCEDIMIENTOS ALAMACENADOS PARA Tbl_BoletaSalario

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Sp_CrearBoletaSal')
DROP PROCEDURE [Sp_CrearBoletaSal]
GO
CREATE PROCEDURE [Sp_CrearBoletaSal]
(
	@idEmpleado int,@monto decimal(28,2)
)
AS
INSERT INTO [dbo].[Tbl_BoletaSalario]
(
	[Bol_id_Salario],[Bol_monto]
)
VALUES
(
	@idEmpleado,@monto
);
GO
```

```
-- DISPARADOR PARA CREAR UNA BOLETA DE SALARIO

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tr_Ins_Boleta_Pago_Planilla')
DROP TRIGGER [Tr_Ins_Boleta_Pago_Planilla]
GO
CREATE TRIGGER [Tr_Ins_Boleta_Pago_Planilla]
ON [dbo].[Tbl_Salario]
FOR INSERT
AS
DECLARE @Hor_precio decimal(28,2);
DECLARE @Hor_normal int;
DECLARE @Hor_extra int;
DECLARE @idSalario int;
DECLARE @Bol_monto decimal(28,2);
SELECT @idSalario = ltrim(rtrim(ins.Sal_id_Salario))
FROM INSERTED ins;
SELECT @Hor_precio = ins.Hor_precio FROM INSERTED ins;
SELECT @Hor_normal = ins.Hor_normal FROM INSERTED ins;
SELECT @Hor_extra = ins.Hor_extra FROM INSERTED ins;
SELECT @Bol_monto = @Hor_normal * @Hor_precio +
(@Hor_extra*(@Hor_precio*1.5));
EXEC [Sp_CrearBoletaSal] @idSalario,@Bol_monto;
GO

-- PROCEDIMIENTOS ALMACENADOS PARA Tbl_Bitacora

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Sp_Ins_Bitacora')
```

```
DROP PROCEDURE [Sp_Ins_Bitacora]
GO
CREATE PROCEDURE  [Sp_Ins_Bitacora]
(
	@usuario varchar(45),
	@servidor varchar(45),
	@datos varchar(max),
	@transac char
)
AS
INSERT INTO [dbo].[Tbl_Bitacora]
(
	[Bit_usuario],[Bit_servidor],[Bit_datos],[Bit_Transac]
)
VALUES
(
	@usuario,@servidor,@datos,@transac
);
GO

-- DISPARADORORES PARA LA TABLE Tbl_Empleado
-- PARA LLEVAR UN REGISTRO DE LOS NUEVOS EMPLEADOS

-- INSERT

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Tr_Ins_Bitacora_InsEmp')
DROP TRIGGER [Tr_Ins_Bitacora_InsEmp]
GO
```

```
CREATE TRIGGER [Tr_Ins_Bitacora_InsEmp]
ON [dbo].[Tbl_Empleado]
FOR INSERT
AS
DECLARE @usuario varchar(45);
DECLARE @servidor varchar(45);
DECLARE @datos varchar(max);
SELECT @usuario = CURRENT_USER;
SELECT @servidor = HOST_NAME();
SELECT @datos = cast(ins.Emp_id_Empleado as varchar(10))+';'+
                    cast(ins.Emp_cedula as varchar(12))+';'+
                    ltrim(rtrim(ins.Emp_nombre))+';'+
                    ltrim(rtrim(ins.Emp_apellido1))+';'+
                    ltrim(rtrim(ins.Emp_apellido2))+';'+
                    ltrim(rtrim(ins.Emp_direccion))+';'+
                    ltrim(rtrim(ins.Emp_contacto))
FROM inserted ins;
EXEC [dbo].[Sp_Ins_Bitacora] @usuario,@servidor,@datos,'I';
GO

-- UPDATE

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Tr_Upd_Bitacora_UpdEmp')
DROP TRIGGER [Tr_Upd_Bitacora_UpdEmp]
GO
CREATE TRIGGER [Tr_Upd_Bitacora_UpdEmp]
ON [dbo].[Tbl_Empleado]
FOR UPDATE
```

```
AS
DECLARE @idEmpleado int;
DECLARE @usuario varchar(45);
DECLARE @servidor varchar(45);
DECLARE @datos varchar(max);
SELECT @usuario = CURRENT_USER;
SELECT @servidor = HOST_NAME();
SELECT @datos = cast(del.Emp_id_Empleado as varchar(10))+';'+
                                cast(del.Emp_cedula as varchar(12))+';'+
                                ltrim(rtrim(del.Emp_nombre))+';'+
                                ltrim(rtrim(del.Emp_apellido1))+';'+
                                ltrim(rtrim(del.Emp_apellido2))+';'+
                                ltrim(rtrim(del.Emp_direccion))+';'+
                                ltrim(rtrim(del.Emp_contacto))
FROM deleted del;
EXEC [dbo].[Sp_Ins_Bitacora] @usuario,@servidor,@datos,'D';
GO

-- DEBIDO A QUE ES UNA BITACORA NO HAY DISPARADOR PARA
-- BORRAR REGISTROS

-- PROCEDIMIENTOS ALMACENADOS PARA Tbl_Cliente

-- INSERT

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Sp_Ins_Cliente')
DROP PROCEDURE [Sp_Ins_Cliente]
GO
```

```
CREATE PROCEDURE [Sp_Ins_Cliente]
(
	@idCliente int,@cedula int,@nombre varchar(45),@apellido1 varchar(45),@apellido2 varchar(45)
)
AS
INSERT INTO [dbo].[Tbl_Cliente]
(
	Cli_id_Cliente,Cli_cedula,Cli_nombre,Cli_apellido1,Cli_apellido2
)
VALUES
(
	@idCliente,@cedula,@nombre,@apellido1,@apellido2
);
GO

-- UPDATE

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Upd_Cliente')
DROP PROCEDURE [Sp_Upd_Cliente]
GO
CREATE PROCEDURE [Sp_Upd_Cliente]
(
	@idCliente int,@cedula int,@nombre varchar(45),@apellido1 varchar(45),@apellido2 varchar(45)
)
AS
UPDATE [dbo].[Tbl_Cliente]
```

```
SET
	Cli_cedula = @cedula,
	Cli_nombre = @nombre,
	Cli_apellido1 = @apellido1,
	Cli_apellido2 = @apellido2
WHERE Cli_id_Cliente = @idCliente;
GO

-- DELETE

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Del_Cliente')
DROP PROCEDURE [Sp_Del_Cliente]
GO
CREATE PROCEDURE [Sp_Del_Cliente]
(
	@idCliente int
)
AS
UPDATE [dbo].[Tbl_Cliente]
SET
	 Cli_estado = 0
WHERE Cli_id_Cliente = @idCliente;
GO

-- PROCEDIMIENTOS ALMACENADOS PARA Tbl_Pelicula

-- INSERT
```

```
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Ins_Pelicula')
DROP PROCEDURE [Sp_Del_Pelicula]
GO
CREATE PROCEDURE [Sp_Ins_Pelicula]
(
	@idPel int,@nombre varchar(45),@censura varchar(45)
)
AS
INSERT INTO [dbo].[Tbl_Pelicula]
(
	Pel_id_Pelicula,Pel_NombrePelicula,Pel_Censura
)
VALUES
(
	@idPel,@nombre,@censura
);
GO

--SELECT * FROM Tbl_Pelicula;
--EXEC [Sp_Ins_Pelicula] 123, 'COCO', 'TODO PÚBLICO';
--EXEC [Sp_Ins_Pelicula] 456, 'LOS CUATRO FANTÁSTICOS', 'TODO PÚBLICO';
--EXEC [Sp_Ins_Pelicula] 789, 'CHUCKY', 'MAYORES 15 AÑOS';
--EXEC [Sp_Ins_Pelicula] 147, 'IT', 'MAYORES 18 AÑOS';
--EXEC [Sp_Ins_Pelicula] 258, 'ANABELLE', 'MAYORES 18 AÑOS';
--EXEC [Sp_Ins_Pelicula] 369, 'LA MUJER MARAVILLA', 'TODO PÚBLICO';

-- UPDATE
```

```
IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Upd_Pelicula')
DROP PROCEDURE [Sp_Upd_Pelicula]
GO
CREATE PROCEDURE [Sp_Upd_Pelicula]
(
	@idPel int,@nombre varchar(45),@censura varchar(45)
)
AS
UPDATE [dbo].[Tbl_Pelicula]
SET
	Pel_id_Pelicula = @idPel,
	Pel_NombrePelicula = @nombre,
	Pel_Censura = @censura
WHERE Pel_id_Pelicula = @idPel
GO

-- DELETE

IF EXISTS(SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Del_Pelicula')
DROP PROCEDURE [Sp_Del_Pelicula]
GO
CREATE PROCEDURE [Sp_Del_Pelicula]
(
	@idPel int
)
AS
UPDATE [dbo].[Tbl_Pelicula]
```

```
SET
	Pel_estado = 0
WHERE Pel_id_Pelicula = @idPel
GO

---PROCEDIMIENTO ALMACENADO PARA Tbl_Sala

-- INSERT

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'Sp_Ins_Butaca')
DROP PROCEDURE [Sp_Ins_Butaca]
GO
CREATE PROCEDURE [Sp_Ins_Butaca](
	@But_id_Butaca int,
	@But_estado char
)
AS
INSERT INTO Tbl_Butaca (But_id_Butaca, But_estado)
VALUES (@But_id_Butaca, @But_estado)
GO

--SELECT * FROM Tbl_Butaca;

--EXEC [SP_INS_BUTACA] 1, 'O';
--EXEC [SP_INS_BUTACA] 2, 'D';
--EXEC [SP_INS_BUTACA] 3, 'O';
--EXEC [SP_INS_BUTACA] 4, 'D';
--EXEC [SP_INS_BUTACA] 5, 'O';
```

```
--EXEC [SP_INS_BUTACA] 6, 'D';
--EXEC [SP_INS_BUTACA] 7, 'O';
--EXEC [SP_INS_BUTACA] 8, 'D';

---PROCEDIMIENTO ALMACENADO PARA Tbl_Sala

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME =
'Sp_Ins_Sala')
DROP PROCEDURE [Sp_Ins_Sala]
GO
CREATE PROCEDURE [Sp_Ins_Sala](
        @Sal_id_Sala int,
        @Sal_id_Butaca int,
        @Sal_id_Pelicula int,
        @Sal_capacidad smallint
)
AS
INSERT INTO Tbl_Sala (Sal_id_Sala, Sal_id_Butaca, Sal_id_Pelicula, Sal_capacidad)
VALUES (@Sal_id_Sala, @Sal_id_Butaca, @Sal_id_Pelicula, @Sal_capacidad)
GO

SELECT * FROM Tbl_Sala;

EXEC [SP_INS_Sala] 1, 1, 123, 50;
EXEC [SP_INS_Sala] 2, 2, 456, 50;
EXEC [SP_INS_Sala] 3, 2, 789, 40;
EXEC [SP_INS_Sala] 4, 3, 147, 55;
EXEC [SP_INS_Sala] 5, 3, 258, 55;
EXEC [SP_INS_Sala] 6, 1, 369, 50;
```

```
GO


----------CURSORES----------


--TABLA PELICULA
--TABLA SALA

--CONOCER LA PELICULA Q ESTA EN CADA SALA
-- DATOS: NOMBRE_PELICULA,FECHA, CENSURA, ID_SALA, CAPACIDAD

---reporte de las peliculas de ccm

IF EXISTS (SELECT NAME FROM DBO.SYSOBJECTS WHERE NAME = 'REPORTE_PASAJEROS')
DROP PROCEDURE [REPORTE_PASAJEROS]
GO
CREATE PROCEDURE [REPORTE_PASAJEROS]
AS
DECLARE @Pel_id_Pelicula CHAR (5), @Pel_NombrePelicula CHAR (45), @Pel_Fecha CHAR (30), @Pel_Censura CHAR(45), @Sal_id_Sala CHAR

---DECLARACION DEL CURSOR Y CONSULTA
DECLARE REPORTE_PELICULAS CURSOR
FOR SELECT P.Pel_id_Pelicula AS ID_PELICULA, P.Pel_NombrePelicula AS NOMBRE_PELICULA, P.Pel_Fecha AS FECHA_Y_HORA, P.Pel_Censura AS CENSURA, S.Sal_id_Sala AS NUMERO_SALA
FROM Tbl_Pelicula P INNER JOIN Tbl_Sala S
```

```
ON P.Pel_id_Pelicula = S.Sal_id_Pelicula

--SE ABRE EL CURSOR, DONDE SE GUARDA TODAS LAS FILAS RESULTANTES DE LA CONSULTA
OPEN REPORTE_PELICULAS

---SE OBTIENE LA PRIMER FILA DEL CURSOR
FETCH REPORTE_PELICULAS INTO @Pel_id_Pelicula, @Pel_NombrePelicula, @Pel_Fecha, @Pel_Censura, @Sal_id_Sala

--- IMPRIMIR CABECERA DEL REPORTE
PRINT 'CODIGO PELÍCULA FECHA / HORA CENSURA #SALA'
PRINT
'-------------------------------------------------------------------------------------------------------------------------'
---AGREGAR LOS DETALLES DEL REPORTE
WHILE @@FETCH_STATUS=0 ---CUANDO YA NO HAY MAS FILAS EL CURSOR SE DETIENE
BEGIN
---SE IMPRIME LOS VALORES DE LAS VARIABLES CON AYUDA DEL CICLO
	PRINT
@Pel_id_Pelicula+SPACE(3)+@Pel_NombrePelicula+SPACE(3)+@Pel_Fecha+SPACE(1)+@Pel_Censura+SPACE(1)+@Sal_id_Sala
	FETCH REPORTE_PELICULAS INTO @Pel_id_Pelicula, @Pel_NombrePelicula, @Pel_Fecha, @Pel_Censura, @Sal_id_Sala
END

---SE CIERRA EL CURSOR
```

```
CLOSE REPORTE_PELICULAS

---SE LIBERA EL CURSOR
DEALLOCATE REPORTE_PELICULAS

EXEC [REPORTE_PASAJEROS]
```