Manual Técnico del Sistema de Punto de Venta para Cine

Manual Técnico

Versión: 1.0 **Fecha:** 22/11/2025 **Sistema:** Sistema de Gestión de Cine

1. Propósito

El propósito de este manual técnico es proporcionar la organización, estructura, configuración y lógica del sistema Cine. Detalla las características físicas y lógicas de los elementos, ofreciendo una guía para el equipo de desarrollo o soporte técnico encargado de administrar, instalar y mantener el sistema.

2. Alcance

El sistema Cine es una aplicación de escritorio (Desktop) diseñada para la gestión de salas de cine. Su objetivo general es generar una herramienta que permita a los administradores crear y gestionar funciones, vender y asignar boletos a asientos, y mantener registros de clientes y empleados.

El manual técnico abarca la descripción de:

1. **Módulos de la Aplicación**: Funcionalidades de Archivo, Ventas, Catálogo, Gestión y Reportes.
2. **Arquitectura Tecnológica**: Uso de Python, PyQt6, y Oracle SQL.
3. **Estructura de la Base de Datos**: Modelo relacional, tablas y secuencias.

3. Tecnologías y Definiciones Importantes

Tecnologías de Desarrollo

- **Backend**: Python y Oracle SQL.
- **Frontend**: Se utiliza la biblioteca **PyQt6** para la interfaz gráfica. (Aunque la propuesta inicial menciona Tkinter, la implementación técnica detallada en el código fuente utiliza PyQt6).
- **Conexión a BD**: Se utiliza la clase `OracleDB` que emplea el módulo `oracledb` de Python para la conexión.

Definiciones Importantes

- **Boleto**: Registro de la venta a un cliente para una butaca y función específicas. Contiene el `precio_pagado` al momento de la compra.
- **Butaca**: Asiento físico dentro de una sala. Sus atributos incluyen `fila`, `numero_butaca` y `tipo_butaca` (ej: normal, VIP, discapacitado).
- **Función**: Proyección de una Película, definida por su `hora_inicio`, `hora_fin`, `precio`, `id_pelicula` e `id_sala`.

- **Pago**: Transacción financiera que registra el `monto` total y el `metodo_pago` (ej: efectivo, tarjeta_credito) asociado a la compra de boletos por un cliente.
- **Cliente**: Entidad que registra datos personales (nombre, apellido, correo obligatorio) y realiza pagos.
- **Empleado**: Personal del cine con credenciales de acceso (`usuario`, `contrasena`) y un `rol` (ej: vendedor, gerente, admin).

4. Descripción de Módulos (Clases y Widgets)

El sistema se estructura alrededor de la clase `MainWindow` de PyQt6 que gestiona la conexión a la base de datos a través de la clase `OracleDB` y organiza las funcionalidades mediante menús.

Módulo de Conexión (OracleDB)

- **Función**: Establece y gestiona la conexión con la base de datos Oracle.
- **Detalles técnicos**: Utiliza la librería `oracledb`. Por defecto, se conecta a `localhost:1521/FREEPDB1` con el usuario `cine` y contraseña `CinePass2024!`. Configura la sesión para usar la zona horaria `'America/Mexico_City'`.

4.1. Módulo Ventas (Vender Boleto y Ver Boletos)

- **Vender Boleto (`VenderBoletoDialog`)**: Es un diálogo (QDialog) que maneja la lógica de la venta de boletos.
  - **Proceso de Entrada**: Recibe `combo_cliente`, `combo_empleado`, `combo_funcion`, `list_butacas` (selección múltiple), y `metodo_pago`.
  - **Lógica de Negocio**: Carga clientes, empleados y funciones disponibles (funciones futuras `WHERE f.hora_inicio > SYSDATE`).
  - **Cálculo de Monto**: El monto (`QDoubleSpinBox`) se actualiza automáticamente basado en la cantidad de butacas seleccionadas y el precio de la función.
  - **Validación de Disponibilidad**: Antes de insertar, verifica que la butaca seleccionada no haya sido vendida para esa función (prevención de condiciones de carrera).
  - **Transacción**: Se crea un registro en la tabla `Pago` y luego se crea un registro en la tabla `Boleto` por cada butaca seleccionada, utilizando las secuencias (`seq_pago`, `seq_boleto`). La transacción se confirma con `self.db.commit()`.
- **Ver Boletos (`BoletosWidget`)**: Muestra el historial de ventas en una tabla (QTableWidget) con 10 columnas, incluyendo ID Boleto, Película, Función Inicio, Sala, Butaca, Cliente y Empleado.

4.2. Módulo Catálogo (CRUD)

Los siguientes módulos utilizan `QWidget` para listar datos y `QDialog` para agregar/editar registros (CRUD):

- **Películas (`PeliculasWidget`)**: Permite administrar (Agregar, Editar, Eliminar, Refrescar) el catálogo de películas. Los campos incluyen Título, Sinopsis, Duración (mín), Idioma, Clasificación y Fecha de estreno.
- **Funciones (`FuncionesWidget`)**: Administra las proyecciones. Los datos clave son `hora_inicio`, `hora_fin`, `precio`, `id_pelicula` e `id_sala`. Requiere que la hora de inicio sea anterior a la hora de fin.
- **Salas (`SalasWidget`)**: Administra las salas de proyección. Los campos son `nombre` y `descripcion_ubicacion`. El nombre es obligatorio.
- **Butacas (`ButacasWidget`)**: Administra los asientos dentro de una sala específica (`id_sala`). Los campos son `fila`, `numero_butaca` y `tipo_butaca`. La clave única es la combinación de `id_sala`, `fila` y `numero_butaca`.

4.3. Módulo Gestión (CRUD de Personas y Pagos)

- **Clientes (`ClientesWidget`)**: Permite la gestión de clientes. Requiere que Nombre, Apellido y Correo sean obligatorios.
- **Empleados (`EmpleadosWidget`)**: Permite la gestión de empleados. Requiere Usuario, Nombre, Apellido y Rol obligatorios. La contraseña es obligatoria para nuevos registros, y se mantiene oculta al editar si no se proporciona una nueva.
- **Pagos (`PagosWidget`)**: Es una interfaz de solo lectura que visualiza los registros de la tabla `Pago`, mostrando ID Pago, Fecha, Monto, Método y el Cliente asociado.

4.4. Módulo Reportes

- **Ventas del día**: Genera un reporte que calcula la suma total del **`monto`** de la tabla `Pago` donde la fecha de pago (`TRUNC(fecha_pago)`) coincide con la fecha actual (`datetime.now().date()`).

5. Modelo Relacional y Distribución de Base de Datos

Base de Datos

El sistema utiliza Oracle Database 21c y gestiona la persistencia de datos a través de la clase `OracleDB`.

Secuencias (Sequences)

Se utilizan secuencias para generar identificadores únicos automáticamente para las tablas principales:

- `seq_boleto`
- `seq_butaca`
- `seq_cliente`
- `seq_empleado`
- `seq_funcion`
- `seq_pago`

- `seq_pelicula`
- `seq_sala`

Tablas Principales

Tabla	Clave Primaria (PK)	Comentarios / Restricciones
Boleto	`id_boleto` (`NUMBER(6)`)	Precio pagado > 0. Registra la compra (Fecha/Precio).
Butaca	`id_butaca` (`NUMBER(10)`)	Restricción única en (`id_sala`, `fila`, `numero_butaca`).
Cliente	`id_cliente` (`NUMBER(12)`)	El correo es único y obligatorio.
Empleado	`id_empleado` (`NUMBER(10)`)	Usuario es único. La contraseña debe almacenarse como *hash*.
Funcion	`id_funcion` (`NUMBER(10)`)	Precio > 0. Registra los horarios de proyección.
Pago	`id_pago` (`NUMBER(10)`)	Monto > 0. Registra `monto` y `metodo_pago`.
Pelicula	`id_pelicula` (`NUMBER(10)`)	`duracion_min` entre 1 y 999.
Sala	`id_sala` (`NUMBER(6)`)	Registra nombre y descripción de ubicación.

Relaciones Clave (Foreign Keys - FK)

Las relaciones de integridad referencial (Foreign Keys) se definen para las transacciones y la estructura física del cine:

- **Boleto** depende de: `Butaca`, `Empleado`, `Funcion`, `Pago`.
- **Butaca** depende de: `Sala`.
- **Funcion** depende de: `Pelicula`, `Sala`.
- **Pago** depende de: `Cliente`.

Procedimientos y Triggers

El código muestra la dependencia de secuencias para la creación de IDs en los procedimientos de guardado (Insert) en los diálogos de Butaca, Empleado, Función, Película y Sala.

La base de datos incluye triggers para la autogeneración de IDs si no se proporcionan, específicamente para las tablas `Pago` y `Cliente`:

- `trg_pago_id`: Asigna `seq_pago.NEXTVAL` a `NEW.id_pago`.
- `trg_cliente_id`: Asigna `seq_cliente.NEXTVAL` a `NEW.id_cliente`.

6. Instalación y Configuración

6.1. Requisitos de Software

Para la instalación y correcta ejecución del sistema Cine:

1. **Base de Datos**: Se requiere una instancia de **Oracle Database** activa (la aplicación asume la configuración por defecto de `FREEPDB1` y el usuario `cine`).
2. **Lenguaje**: **Python 3.x**.
3. **Librerías de Python**:
    - `oracledb`: Para la conectividad con la base de datos Oracle.
    - `PyQt6`: Para la ejecución de la interfaz gráfica y sus componentes (widgets, diálogos).

6.2. Configuración de la Base de Datos

1. **Creación de Esquema y Objetos**: Se deben crear las secuencias (`seq_*`) y las tablas (`Boleto`, `Butaca`, `Cliente`, `Empleado`, `Funcion`, `Pago`, `Pelicula`, `Sala`), junto con sus restricciones, índices y *foreign keys*.
2. **Configuración de Triggers**: Los triggers `trg_pago_id` y `trg_cliente_id` deben estar activos para la correcta generación automática de IDs para pagos y clientes.
3. **Ajuste de Zona Horaria**: La conexión configura explícitamente la sesión de Oracle para usar la zona horaria `'America/Mexico_City'`.

6.3. Configuración de Conexión en la Aplicación

La conexión se maneja dentro de la clase `OracleDB` en `main.py`. Si se requiere modificar los parámetros de conexión (usuario, contraseña, host, puerto o servicio), estos deben ajustarse en la inicialización de la clase `OracleDB` dentro de la función `__init__` de `MainWindow`.

7. Descripción de Usuarios

Aunque las fuentes no proporcionan una lista exhaustiva de privilegios o roles de acceso al sistema, la aplicación está diseñada para manejar diferentes roles de usuario, lo cual impacta en la lógica de negocio (por ejemplo, al vender un boleto se registra el `id_empleado`).

- **Entidad Gestionada**: Los usuarios del sistema son administrados en el módulo de Empleados, que registra el `usuario` y el `rol`.

- **Roles Esperados**: Ejemplos de roles definidos son `vendedor`, `gerente`, `admin`, u `operador`.
- **Funcionalidades de Rol**: La mayoría de las acciones (CRUD) y la venta de boletos requieren que el usuario esté conectado a la base de datos y sea un `Empleado` registrado.

El módulo de Empleados obliga a registrar el Usuario, Nombre, Apellido y Rol. La contraseña debe almacenarse de forma segura (como *hash*).